Re-evaluating the impact of data cleaning on ML classification

(부제) Anaysis of Data cleaning effect based on Noise Level in Datasets

분산 클라우드 컴퓨팅 – 팀 프로젝트 2차발표

데이터사이언스학과 황선진
자동차공학과 김진우
데이터사이언스학과 야오와말

목차

1 프로젝트 배경

- 프로젝트 관련 논문
  - CleanML: A Study for Evaluating the Impact of Data Cleaning on ML Classification Tasks (2021, ICDE)
    - (내용) ML 분류 모델 성능에 대한 데이터 정제 영향을 체계적·실험적 분석
    - (의의) 데이터 정제가 ML 분류 성능 향상에 효과 : 실험적 파악, 향후 데이터 정제 연구의 기반 마련

✓ 오류 유형(5가지)

- Missing Values(결측치)
- Outliers (이상치)
- Mislabels(라벨링 오류)
- Inconsistencie(일관성 오류)
- Duplicates(중복값)

데이터 정제 →

✓ 분류 모델(7가지)

- Linear Regression
- Discision Tree
- Random Forest
- Adaboost
- XGBoost
- KNN (k- Nearest Neighbors)
- Naive Bayes

1 프로젝트 배경

- 프로젝트 관련 논문 : 한계점
  - 데이터 정제 효과가 데이터셋에 의존(depend on Dataset) 함을 파악하였으나 데이터셋의 오류 유형만 언급했을 뿐, 구체적인 오류 데이터 비율·관련 추가 실험이 존재하지 않음

✓ 논문 발췌 : Overall Observations summary

TABLE 16. Summary of Empirical Findings for Single Error Types

Error Type	Impact on ML	Does the impact depend on			
		Datasets	Scenarios	Cleaning Algos	ML Algorithms
Duplicates	Varying (Mostly S & N)		No	Yes	No
Inconsistencies	Varying (Mostly S)		No	N.A.	No
Missing Values	Varying (Mostly P & S)	Yes	No	Yes	No
Mislabels	Varying (Mostly P & S)		Yes	N.A.	No (except Boosting)
Outliers	Varying (Mostly S)		No	Yes	No (except KNN)

※ Strong Dependency on Dataset

the cleaning impact depends on datasets — while two datasets may contain errors of the same type, the distributions of those errors can be vastly different. Therefore, practitioners should never make arbitrary cleaning decisions dealing with dirty data in ML classification tasks.

✓ 논문 발췌 : Datasets

We collected 14 real-world datasets with varying error types and error rates

TABLE 3. Dataset and Error Types

Datasets	Error Types				
	Inconsistencies	Duplicates	Missing Values	Outliers	Mislabels
Citation		x			
EEG				x	x
Marketing			x		x
Movie	x	x			
Company	x				
Restaurant	x	x			
Sensor				x	
Titanic			x		x
Credit			x	x	
University	x				
USCensus			x		x
Airbnb		x	x	x	
BabyProduct			x		
Clothing					x

※ 단, 오류 유형이 아닌 오류 비율에 대한 구체적인 언급은 없음

1 프로젝트 배경

- 문제의식에 기반한 아이디어 제안
  - (문제의식) 해당 논문에서는 Error rates(오류 데이터 비율)과 관련된 실험이 존재하지 않음
  - (아이디어 제안) Error rates(오류비율) 수준에 따른 데이터 정제효과 분석

✓ 기존 논문(Clean ML)

Dataset Error rates 고려 X

✓ 프로젝트 제안(Our project)

Dataset Error rates 조정

2 프로젝트 방법론

- 데이터 오류 비율 조정/정제 – ML 모델성능 평가 실험
  - (참고 코드) CleanML 논문의 공개 코드 : https://github.com/chu-data-lab/CleanML
  - 오류유형 중 Missing Values(결측치), Outliers(이상치), Duplicates(중복값) 선택하여 실험

※ (선택 이유) 데이터 분석 시 자주 접하는 오류유형이며, 다양한 오류 탐지 및 정제 방법 적용 가능

* 다른 오류 유형(Mislabels(라벨링 오류), Inconsistencie(일관성 오류))의 경우 탐지 및 정제 방법이 제한적

  - 논문의 실험 데이터셋 중 Airbnb 데이터셋 선택 : 선택한 3가지 오류유형 모두 포함

✓ 논문 발췌 : 오류유형에 따른 탐지 및 정제 방법

TABLE 2. Automatic Cleaning Methods

Error Type	Detection Method	Repair Method
Missing Values	Empty Entries	Deletion
		Mean_Mode, Mean_Dummy Median_Mode, Median_Dummy Mode_Mode, Mode_Dummy
		HoloClean
Outliers	SD	Mean, Median, Mode
	IQR	
	IF	HoloClean
Duplicates	Key Collision	Deletion
	ZeroER	
Inconsistencies	OpenRefine	Merge
Mislabels	cleanlab	cleanlab

✓ 논문 발췌 : Datasets and error types

TABLE 3. Dataset and Error Types

Datasets	Error Types				
	Inconsistencies	Duplicates	Missing Values	Outliers	Mislabels
Citation		x			
EEG				x	x
Marketing			x		x
Movie	x	x			
Company	x				
Restaurant	x	x			
Sensor				x	
Titanic			x		x
Credit			x	x	
University	x				
USCensus			x		x
Airbnb		x	x	x	
BabyProduct			x		
Clothing					x

2 프로젝트 방법론

- 데이터 오류 비율 조정 - 데이터 정제 – ML 모델성능 평가 실험
  - 프로세스 : ❶ - ❸ 반복 수행 (오류유형마다 구체적인 방법은 상이함)
  - (결과 분석) 데이터 정제 전/후 성능 지표 비교·분석 : 정확도 지표로 활용

❶ 데이터 오류 비율 조정 → ❷ 데이터 정제 수행 → ❸ ML 모델 성능 평가 및 평가지표 값 저장

- Missing Values(결측치)
- Outliers (이상치)
- Duplicates(중복값)

- Mean
- Median
- Mode
- Deletion

- Linear Regression
- Discision Tree
- Random Forest
- Adaboost
- XGBoost
- KNN (k- Nearest Neighbors)
- Naive Bayes

3 실험결과

1. Missing Values(결측치)

- 데이터의 결측치를 추가적으로 늘려 모델 성능 비교
- [프로세스] 학습 데이터(원본, 결측치 1.5배, 결측치 2배) → 모델 학습 → 원본 테스트셋으로 평가

  * (결측치 수) 16,537개 → 24,802개 → 33,074개

  ✓ 결측치가 증가함에 따라 단순 삭제보다 정제할 경우 모델 성능이 향상되는 추세

  ✓ 정제방법(mean, median, mode)에 따른 차이는 크지 않음

  ✓ 특정 알고리즘에서 결측치 정제가 더 나은 성능을 가져오는 경향

결측값 수준	적음(원본)					중간(원본 * 1.5배)					중간(원본 * 2배)				
	삭제	정제				삭제	정제				삭제	정제			
		평균	mean	median	mode		평균	mean	median	mode		평균	mean	median	mode
LR	0.671	0.671	0.671	0.671	0.671	0.672	0.673	0.674	0.674	0.673	0.674	0.673	0.673	0.672	0.672
DT	0.679	0.680	0.678	0.680	0.681	0.667	0.677	0.675	0.680	0.677	0.645	0.674	0.677	0.677	0.668
KNN	0.661	0.661	0.661	0.661	0.661	0.655	0.652	0.651	0.651	0.653	0.646	0.652	0.652	0.652	0.652
RF	0.745	0.741	0.742	0.743	0.739	0.725	0.741	0.742	0.739	0.741	0.703	0.735	0.732	0.737	0.735
Ada	0.684	0.684	0.684	0.684	0.684	0.684	0.688	0.688	0.688	0.688	0.686	0.686	0.685	0.685	0.687
NB	0.639	0.639	0.639	0.639	0.639	0.638	0.644	0.644	0.643	0.645	0.640	0.643	0.643	0.643	0.644

3 실험결과

2. Outliers (이상치)

- (첫번째 방법) : 이상치 탐지 기준을 조정하여 이상치를 정제해 모델 성능 비교 수행
- [프로세스] 기존 학습 데이터(원본), 이상치 기준 변경 학습 데이터 → 모델 학습 → 각각의 테스트셋으로 평가

※ 논문에 기재되어있는 시나리오(test on Cleaned Test Set)

* (이상치 기준 변경에 따른 이상치 개수, SD 기준) [적음] 1,772개 → [중간] 3,784개 → [많음] 34,918개

✓ 이상치 기준 변경/정제에 따른 모델 성능 차이는 거의 미미함

✓ 특정 알고리즘에서 이상치 기준 변경/정제에 대해 조금 더 민감하게 반응하는 경향

※ 성능 : 정제방법(Mean, Median, Mode) 평균

이상치수준	많음	중간	적음	-	적음	중간	많음
모델 / 이상치 기준	SD: std*1.5	SD: std*3	SD: std*4	원본	IQR * 2.0	IQR * 1.5	IQR * 1.0
LR	0.675	0.673	0.674	0.674	0.673	0.674	0.674
DT	0.688	0.690	0.693	0.691	0.690	0.688	0.686
KNN	0.663	0.664	0.664	0.664	0.664	0.664	0.664
RF	0.748	0.748	0.747	0.747	0.750	0.749	0.748
Ada	0.687	0.687	0.689	0.688	0.684	0.687	0.689
NB	0.645	0.644	0.649	0.650	0.639	0.644	0.644

3 실험결과

2. Outliers (이상치)

- (두번째 방법) : 이상치를 정제할 데이터 비율을 조정하여 모델 성능 비교 수행
- [프로세스] 이상치 정제 데이터 비율(10, 25, 30, 50, 100%(clean)) → 모델 학습 → 각각의 테스트셋으로 평가

✓ 이상치 정제 비율에 따른 모델 성능 - 일관적인 추세는 보이지 않음

✓ 특정 알고리즘에서 전체 데이터를 정제하였을 때 성능이 소폭 개선되는 모습을 보임

	낮음	← 정제수준 →			높음	낮음	← 정제수준 →			높음	낮음	← 정제수준 →			높음
Model	SD - Mean					SD - Median					SD - Mode				
	10%	25%	30%	50%	100%	10%	25%	30%	50%	100%	10%	25%	30%	50%	100%
Decision Tree	0.673	0.679	0.683	0.679	0.673	0.679	0.679	0.673	0.679	0.675	0.680	0.677	0.676	0.679	0.685
Random Forest	0.741	0.736	0.710	0.713	0.742	0.744	0.734	0.706	0.708	0.744	0.743	0.737	0.710	0.714	0.743
Adaboost	0.680	0.685	0.686	0.679	0.681	0.682	0.683	0.683	0.683	0.682	0.683	0.682	0.682	0.682	0.682
Logistic Regression	0.610	0.615	0.611	0.609	0.615	0.606	0.607	0.609	0.609	0.608	0.611	0.614	0.611	0.611	0.612
KNN	0.657	0.653	0.653	0.655	0.667	0.655	0.653	0.652	0.655	0.673	0.655	0.654	0.654	0.656	0.673
Model	IQR - Mean					IQR - Median					IQR - Mode				
	10%	25%	30%	50%	100%	10%	25%	30%	50%	100%	10%	25%	30%	50%	100%
Decision Tree	0.671	0.666	0.675	0.675	0.672	0.674	0.679	0.678	0.677	0.676	0.67	0.676	0.679	0.685	0.673
Random Forest	0.736	0.73	0.708	0.71	0.738	0.732	0.699	0.697	0.7	0.737	0.732	0.704	0.703	0.696	0.736
Adaboost	0.686	0.688	0.681	0.679	0.681	0.684	0.683	0.683	0.684	0.683	0.679	0.68	0.681	0.681	0.68
Logistic Regression	0.621	0.623	0.617	0.611	0.623	0.619	0.618	0.622	0.613	0.616	0.618	0.618	0.618	0.614	0.615
KNN	0.657	0.651	0.651	0.656	0.671	0.658	0.653	0.654	0.652	0.668	0.653	0.65	0.652	0.651	0.667

3 실험결과

3. Duplicates(중복값)

- 중복값 정제 후 오버샘플링을 적용하여 모델 성능 비교 수행 : 중복값 정제를 통한 augmentation 효과 파악
- (프로세스) 원본(Dirty)과 중복값 정제 데이터(Clean, AutoER)에서 오버샘플링을 적용 → 원본 테스트셋으로 평가

  * (오버샘플링 비율, Y:N) 6:4 → 5.5:4.5 → 5:5

  ✓ (각 데이터셋별 오버샘플링 효과) 대체로 오버샘플링이 많아질수록 성능 하락, 과적합 경향

	Dirty				Clean (경/위도만 활용)				AutoER (경위도 + a)			
비율	기존	오버샘플링			기존	오버샘플링			기존	오버샘플링		
		6:4	5.5:4.5	5:5		6:4	5.5:4.5	5:5		6:4	5.5:4.5	5:5
Logistic regression	0.798	0.785	0.753	0.697	0.798	0.789	0.756	0.698	0.797	0.794	0.784	0.770
KNN classification	0.794	0.767	0.761	0.754	0.798	0.769	0.761	0.755	0.798	0.777	0.766	0.763
Decision tree classification	0.791	0.761	0.757	0.762	0.795	0.767	0.768	0.761	0.793	0.766	0.772	0.761
Adaboost classification	0.802	0.785	0.751	0.700	0.802	0.786	0.752	0.708	0.802	0.791	0.781	0.774
Random forest classification	0.801	0.811	0.805	0.799	0.799	0.813	0.804	0.801	0.797	0.814	0.812	0.807
Guassian naïve bayes	0.040	0.024	0.023	0.023	0.040	0.024	0.024	0.022	0.040	0.024	0.023	0.025
XGBoost	0.795	0.805	0.801	0.796	0.802	0.805	0.801	0.803	0.797	0.803	0.807	0.803

3 실험결과

3. Duplicates(중복값)

- 중복값 정제 후 오버샘플링을 적용하여 모델 성능 비교 수행 : 중복값 정제를 통한 augmentation 효과 파악
- (프로세스) 원본(Dirty)과 중복값 정제 데이터(Clean, AutoER)에서 오버샘플링을 적용 → 원본 테스트셋으로 평가

  * (오버샘플링 비율, Y:N) 6:4 → 5.5:4.5 → 5:5

✓ (각 데이터셋 동일 비율 별 오버샘플링 효과) 대체로 AutoER > Clean > Dirty 순으로 정제 후 효과 향상

	Dirty				Clean (경/위도만 활용)				AutoER (경위도 + a)			
비율	기존	오버샘플링			기존	오버샘플링			기존	오버샘플링		
		6:4	5.5:4.5	5:5		6:4	5.5:4.5	5:5		6:4	5.5:4.5	5:5
Logistic regression	0.798	0.785	0.753	0.697	0.798	0.789	0.756	0.698	0.797	0.794	0.784	0.770
KNN classification	0.794	0.767	0.761	0.754	0.798	0.769	0.761	0.755	0.798	0.777	0.766	0.763
Decision tree classification	0.791	0.761	0.757	0.762	0.795	0.767	0.768	0.761	0.793	0.766	0.772	0.761
Adaboost classification	0.802	0.785	0.751	0.700	0.802	0.786	0.752	0.708	0.802	0.791	0.781	0.774
Random forest classification	0.801	0.811	0.805	0.799	0.799	0.813	0.804	0.801	0.797	0.814	0.812	0.807
Guassian naïve bayes	0.040	0.024	0.023	0.023	0.040	0.024	0.024	0.022	0.040	0.024	0.023	0.025
XGBoost	0.795	0.805	0.801	0.796	0.802	0.805	0.801	0.803	0.797	0.803	0.807	0.803

3 실험결과

3. Duplicates(중복값)

- 중복값 정제 후 오버샘플링을 적용하여 모델 성능 비교 수행 : 중복값 정제를 통한 augmentation 효과 파악
- (프로세스) 원본(Dirty)과 중복값 정제 데이터(Clean, AutoER)에서 오버샘플링을 적용 → 원본 테스트셋으로 평가

  * (오버샘플링 비율, Y:N) 6:4 → 5.5:4.5 → 5:5

  ✓ 가장 좋은 성능을 보이는 Random forest 모델의 오버샘플링 효과

  - 특정 비율까지 오버샘플링할 때 성능이 개선
  - 또한, 해당 알고리즘이 중복값에 대해 가장 덜 영향을 받는 경향

Dataset	Dirty				Clean				AutoER			
Ratio	0.678	0.6	0.55	0.5	0.672	0.6	0.55	0.5	0.676	0.6	0.55	0.5
Random forest classification	0.801	0.811	0.805	0.799	0.799	0.813	0.804	0.801	0.797	0.814	0.812	0.807

Ratio	원본			0.6			0.55			0.5		
Dataset	Dirty	Clean	AutoER	Dirty	Clean	AutoER	Dirty	Clean	AutoER	Dirty	Clean	AutoER
Random forest classification	0.801	0.799	0.797	0.811	0.813	0.814	0.805	0.804	0.812	0.799	0.801	0.807

4 프로젝트 결론

- (정리) 데이터 오류 비율 조정 및 정제 적용 후의 ML 모델 성능 평가
  - CleanML 논문의 한계점인 오류 비율 관련 연구 부족 : 관련 실험 프로젝트 수행

✓ 프로젝트 실험결과 정리

1. 결측치 : 결측값 수준이 높아질수록 정제에 따른 모델 성능이 향상되는 경향
2. 이상치 : 이상치 기준 및 정제비율 조정은 모델 성능과 관련성이 낮음
3. 중복값 : 중복값 정제 후 오버샘플링을 적용할 시 모델 성능 하락을 소폭 개선 가능

- (기대효과) 데이터 분석가의 직관에 의존하는 데이터 정제에 실증적인 연구 제공
  - 데이터 오류 비율은 쉽게 파악 가능하기 때문에 실제 데이터 분석에 활용 가능

- (한계점 및 후속 연구)
  - 활용한 데이터셋 외에 다양한 데이터셋 적용 및 분석 필요
  - 오류비율 조정 외에도 ML 회귀모델, 딥러닝 모델, 데이터 증강 적용 등 추가 연구필요

감사합니다

❖ (실험 담당자)

결측값, 이상치(1) : 황선진
중복값 : 김진우
이상치(2) : 야오와말