최종 발표

7조

김희균, 송준규, 정승연, 정하연

목차

1. Goal of the Project
2. 데이터 분석 및 전처리
    - Get the Data
    - Discover and Visualize the data
    - Prepare the data
3. Select and Train Models
4. Final Test
5. Feedback

1. Goal of the Project

“ 높은 점수보다 Overfitting을 막는 최대한 내성이 강한 모델을 만들자”

(노이즈, 테두리, shifted)

2. 데이터 분석 및 전처리

(1) 15개(0~9, +, -, /, x, =) 클래스 이외의 데이터 → **제거**

	숫자	기호
Training	(15119, 28, 28)	(15329, 28, 28)
Test	(2160, 28, 28)	(2190, 28, 28)

〈Data Cleaning 이전 dataset의 shape〉

	숫자 + 기호
Training	(30448, 28, 28)
Test	(4350, 28, 28)

〈15개 클래스 분류기를 위한 데이터 통합〉

최종 데이터셋 shape	
Training	(26249, 28, 28)
Test	(3730, 28, 28)

〈최종 데이터셋〉

제거된 데이터셋 개수	
Training	4199 (13.8%)
Test	620 (14.3%)

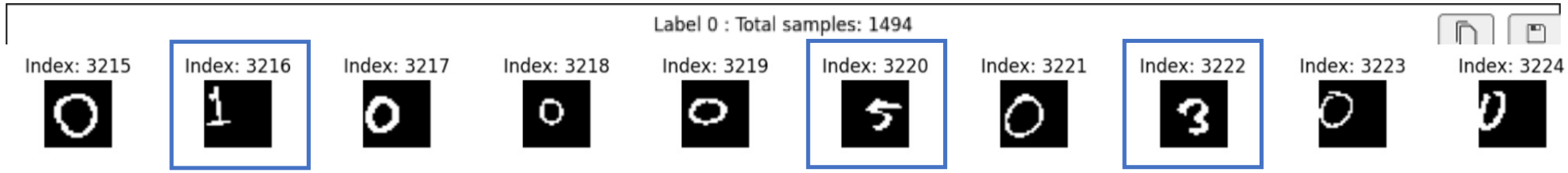
〈15개 이외 라벨 제거〉

2. 데이터 분석 및 전처리

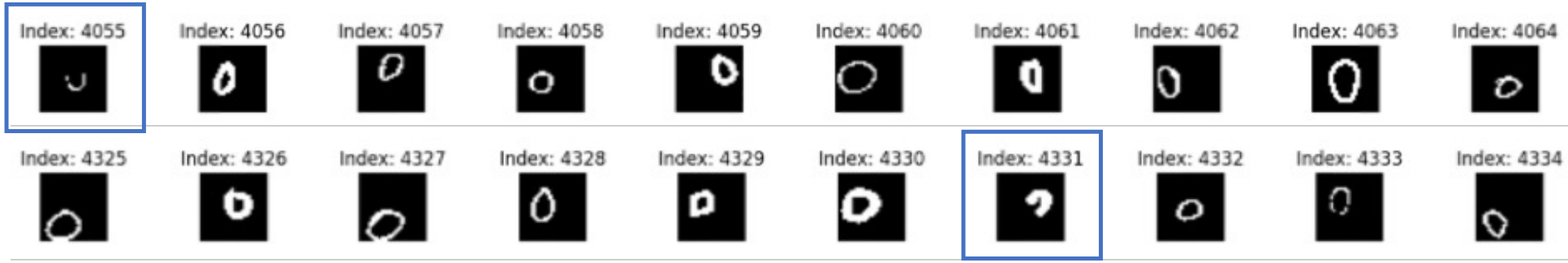
- Discover and Visualize the data

(1) 15개(0~9, +, -, /, x, =) 클래스 이외의 데이터 → **제거**

(2) 잘못된 라벨링 → **육안으로 판별 후 제거**



(3) 학습에 애매한 데이터 → **육안으로 판별 후 제거 (2, 3번 통합 1700여 개 제거됨)**



2. 데이터 분석 및 전처리

- Discover and Visualize the data

(4) Handmade dataset 데이터 수 : 약 30000개 → **Combined Dataset 구성 (총 60,000개, trainset : 각 4200개, testset : 각 700개)**

(5) 픽셀값(feature 값)의 차이 → **파이프라인에 Normalizer( ) 추가 (Normalizer에 대한 분석 및 결정 과정 추가 필요)**

- Original Dataset : 0~255
- Handmade Dataset : 0~1

(6) 노이즈가 있는 데이터 → **노이즈 추가하여 trainset, testset 구축 + 파이프라인에 denoising 함수 추가**

- 픽셀값이 0에서 1이라고 하면, 0에서 0.4 사이의 랜덤 값을 픽셀에 추가
- 임계값을 0.4로 설정하여 픽셀값이 0.4보다 크면 유효한 값(숫자나 기호)으로 판별

2. 데이터 분석 및 전처리

- Discover and Visualize the data

shift된 데이터, 테두리가 남은 데이터 → **파이프라인에 중앙화, 최대화 과정 추가**

(7) 비슷한 숫자 및 기호의 존재 → **Confusion Matrix 분석 계획**

(8) 데이터 크기 : 28*28 데이터만 있음 → **현재 가진 데이터를 확대, 축소해서 새로운 데이터를 만드는 데 한계가 있음**

2. 데이터 분석 및 전처리

- Prepare the data

Original Dataset 구성

Training/validation

Handmade Dataset 구성

Training/validation

Combined Dataset 구성

Training/validation

Final Test Dataset 구성

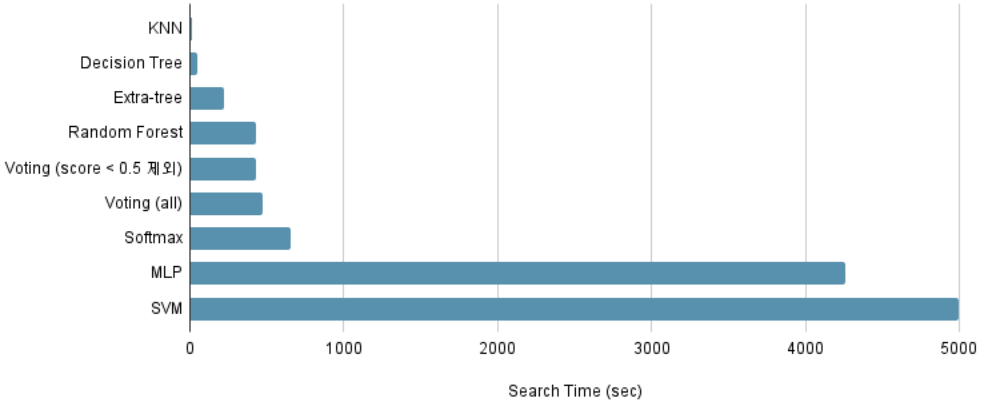
3. Select and Train Models

(1) Original Dataset

Model	KNN	SVM	Extra-tree	Softmax	Decision Tree	Random Forest	MLP	Voting (score < 0.5 제외)	Voting (all)
Search Time (sec)	16.10	takes too long	218.40	654.03	51.40	426.11	4255.92	431.	472.67
Best Parameters	**n_neighbors : 3**		max_depth : 20 **n_estimators : 300**	C : 0.1 **max_iter : 500** multi_class : multinomial	Criterion : gini **max_depth : 50**	**n_estimators : 900**	alpha : 0.1 **max_iter : 500**	**soft**	**hard**
Best Accuracy on Test Dataset	0.6059		0.7263	0.3769	0.4903	0.7445	0.7370	soft: 0.7499, hard: 0.7458	soft: 0.7190, hard: 0.7357

3. Select and Train Models

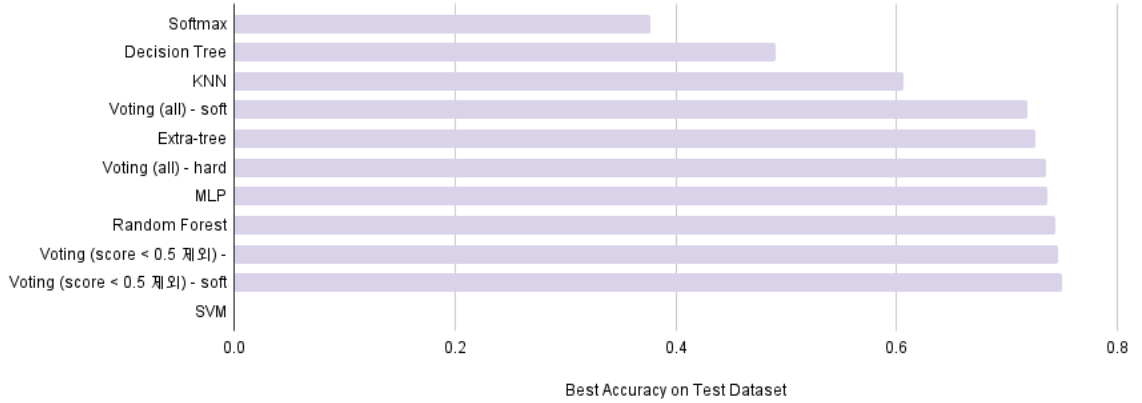
- Search Time (sec)



→ Original, Our's 차이점 분석

→ 최종 모델 및 결정 이유

- Accuracy : Original Testset



- Accuracy : Our Team's Dataset

4. Fine-Tune the Model

(2) Handmade Dataset

Model	KNN	SVM	Extra-tree	Softmax	Decision Tree	Random Forest	MLP	Voting (all)
Search Time (sec)	19.91	5805.91	114.86	376.11	104.96	7228.69	25832.11	691.17 1540.86
Best Parameters	**n_neighbors : 5**	**C : 5**	max_depth : 20 **n_estimators : 300**	C : 1 **max_iter : 500** multi_class : multinomial	Criterion : entropy **max_depth : 50**	**n_estimators : 700**	alpha : 0.01 **max_iter : 500**	**soft**
Best Accuracy on Test Dataset	0.8925	0.9341	0.9115	0.8374	0.7448	0.8990	0.9120	soft: 0.9196, hard: 0.9168

4. Optimize the Model

- Search Time (sec)

→ Overfitting 여부 분석

- Accuracy : Original Testset

- Accuracy : Our Team's Dataset

4. Fine-Tune the Model

- Combined Dataset

Model	KNN	SVM	Extra-tree	Softmax	Decision Tree	Random Forest	MLP	Voting (all)
Search Time (sec)								
Best Parameters								
Best Accuracy on Test Dataset								

4. Fine-Tune the Model

[최적화 과정 제시]

1. 최대화, 최적화 과정 추가

[결과 분석]

- 학습 시간
- 예측 시간 (inference time)
- 정확도

[epoch에 따른 Learning curve]

- 여기서 파라미터 별 learning curve 다 따로 제시해야 하나?

4. Fine-Tune the Model

- Search Time (sec)

- Accuracy : Original Testset

- Accuracy : Our Team's Dataset

→ 성능 분석

4. Final Test

Confusion Matrix

Overfitting 분석

4. Final Test

제공된 Test dataset으로
최종 평가

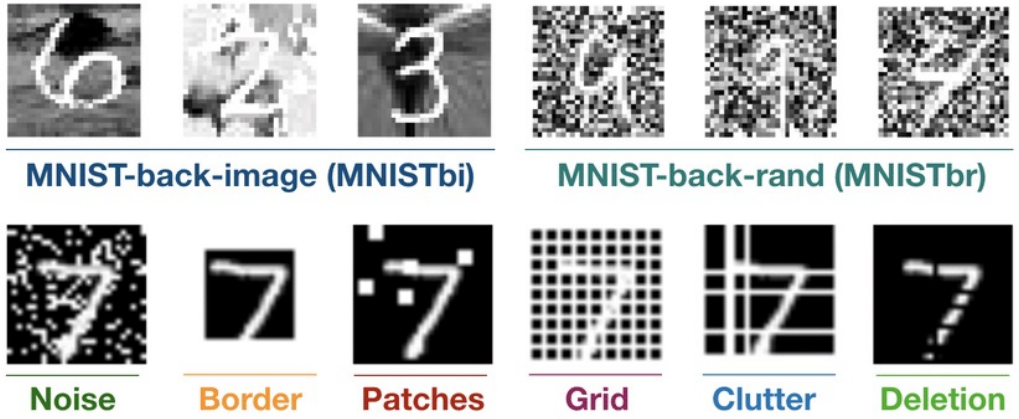
5. Feedback

- Shifted data 처리 방식 결정 과정

- 한계 1 : Rotate, 테두리

5. Feedback

- 한계 2 : 다양한 노이즈에 대한 해결책?



감사합니다