

3장 분류

감사의 글

자료를 공개한 저자 오렐리앙 제롱에게 깊은 감사를 드립니다. 이와 더불어 한빛미디어로부터 강의준비에 필요한 자료를 지원받았음을 밝히며, 이에 대해 진심어린 감사를 전합니다.

목차

1. MNIST
2. 이진 분류기 훈련
3. 성능 측정
4. 다중 클래스 분류
5. 에러 분석
6. 다중 레이블 분류
7. 다중 출력 분류

3.1 MNIST

MNIST 데이터셋

- 미국 고등학생과 인구조사국 직원들이 손으로 쓴 70,000개의 숫자 이미지로 구성된 데이터셋
- 사용된 0부터 9까지의 숫자는 각각 $28*28=784$ 크기의 픽셀로 구성된 이미지 데이터
 - 2차원 어레이가 아닌 길이가 784인 1차원 어레이로 제공
- 레이블: 총 70,000개의 사진 샘플이 표현하는 값

5	0	4	1	9	2	1	3	1	4
3	5	3	6	1	7	2	8	6	9
4	0	9	1	1	2	4	3	2	7
3	8	6	9	0	5	6	0	7	6
1	8	1	9	3	9	8	5	9	3
3	0	7	4	9	8	0	9	4	1
4	4	6	0	4	5	6	1	0	0
1	7	1	6	3	0	2	1	1	7
8	0	2	6	7	8	3	9	0	4
6	7	4	6	8	0	7	8	3	1

문제 정의

- 지도학습: 각 이미지가 담고 있는 숫자가 레이블로 지정됨.
- 분류: 이미지 데이터를 분석하여 0부터 9까지의 숫자로 분류
 - 이미지 그림을 총 10개의 클래스로 분류하는 다중 클래스 분류(multiclass classification)
 - 다항 분류(multinomial classification)라고도 불림
- 배치 또는 온라인 학습: 둘 다 가능
 - 모델에 따라 처리 방법이 다름
 - 확률적 경사하강법(stochastic gradient descent, SGD): 배치와 온라인 학습 모두 지원
 - 랜덤 포레스트 분류기: 배치 학습

훈련 셋과 데이터 셋 나누기

- MNIST 데이터셋 이미 6:1 분류되어 있음
- 훈련 셋: 앞쪽 60,000개 이미지
- 테스트 셋: 나머지 10,000개의 이미지

3.2 이진 분류기 훈련

예제: 5-감지기

- 이미지 샘플이 5를 표현하는지 여부를 판단하는 이진 분류기
- 이를 위해 레이블을 0 또는 1로 수정
 - 기존에 0부터 9까지의 숫자 대신 0, 1 두 값으로만 구성된 새로운 타깃 벡터를 생성해서 학습에 사용
 - 숫자 5를 가리키는 이미지 레이블: 1
 - 숫자 5 이외의 수를 가리키는 이미지 레이블: 0

SGD 분류기 활용 학습

- SGDClassifier(SGD 분류기)
 - 확률적 경사 하강법(stochastic gradient descent) 사용
 - 한 번에 하나씩 훈련 샘플 처리 후 파라미터 조정
 - 매우 큰 데이터셋 처리에 효율적이며 온라인 학습에도 적합함.

- 학습

```
from sklearn.linear_model import SGDClassifier
```

```
sgd_clf = SGDClassifier(max_iter=1000, tol=1e-3, random_state=42)  
sgd_clf.fit(X_train, y_train_5)
```

3.3 성능 측정

성능 측정 세가지 방법

- 교차 검증을 활용한 정확도 측정
- 정밀도/재현율 조율
- AUC 측정

교차 검증을 사용한 정확도 측정

- 2장에서 배운 교차검증 기술을 이용하여 SGD 분류기의 성능을 측정
- 성능 측정 기준: 정확도

- 정확도: 전체 샘플을 대상으로 정확하게 예측한 비율
 - 여기서는 숫자 5를 표현하는 이미지를 True로 예측한 비율

```
from sklearn.model_selection import cross_val_score
```

```
cross_val_score(sgd_clf, X_train, y_train_5, cv=3, scoring="accuracy")
```

- 교차 검증 결과가 95% 이상으로 매우 우수한 것으로 나옴.
- 하지만 무조건 '5 아님'이라고 찍는 분류기도 90%의 정확도를 보임.
- 훈련 세트의 샘플이 불균형적으로 구성되었다면, 정확도를 분류기의 성능 측정 기준으로 사용하는 것은 피해야 함

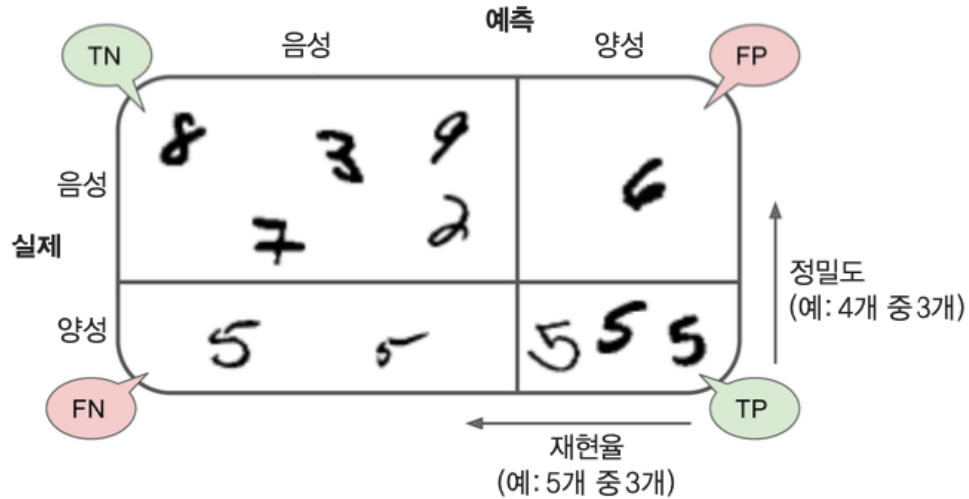
오차 행렬, 정밀도, 재현율

- 오차 행렬을 조사하여 분류기의 성능을 평가할 수 있음
- 정밀도와 재현율 확인

오차 행렬(confusion matrix)

- 오차 행렬: 클래스 별 예측 결과를 정리한 행렬
- 오차 행렬의 행은 실제 클래스를, 열은 예측된 클래스를 가리킴
 - 클래스 A의 샘플이 클래스 B의 샘플로 분류된 횟수를 알고자 하면 A행 B열의 값을 확인
- 예제: 숫자 5의 이미지 샘플을 3으로 잘못 예측한 횟수를 알고 싶다면?
 - 6행 4열, 즉, (6,4) 인덱스에 위치한 값을 확인 (0부터 9까지의 숫자임에 주의)

- 예제: '숫자 5-감지기'에 대한 오차 행렬 생성
 - 결과는 2x2 모양의 2차원 어레이
 - 레이블의 값이 0과 1 두 개의 값으로 구성되기 때문



정밀도(precision)

- 양성 예측의 정확도
- 여기서는 숫자 5라고 예측된 값들 중에서 진짜로 5인 숫자들의 비율

$$\text{precision} = \frac{TP}{TP + FP} = \frac{3530}{3530 + 687} = 0.8370$$

- 정밀도 하나만으로 분류기의 성능을 평가할 수는 없음
 - 숫자 5를 가리키는 이미지 중에 숫자 5가 아니라고 판명한 경우를 함께 고려하지 않기 때문
- 분류기가 정확하게 예측한 양성 샘플의 비율인 재현율을 함께 다루어야 함

재현율(recall)

- 양성 샘플에 대한 정확도, 즉, 분류기가 정확하게 감지한 양성 샘플의 비율
- 재현율을 민감도(sensitivity) 또는 참 양성 비율(true positive rate)로도 부름

$$\text{recall} = \frac{TP}{TP + FN} = \frac{3530}{3530 + 1891} = 0.6512$$

F_1 점수(F_1 score)

- 정밀도와 재현율의 조화 평균:

$$F_1 = \frac{2}{\frac{1}{precision} + \frac{1}{recall}} = \frac{2 \cdot precision \cdot recall}{precision + recall}$$

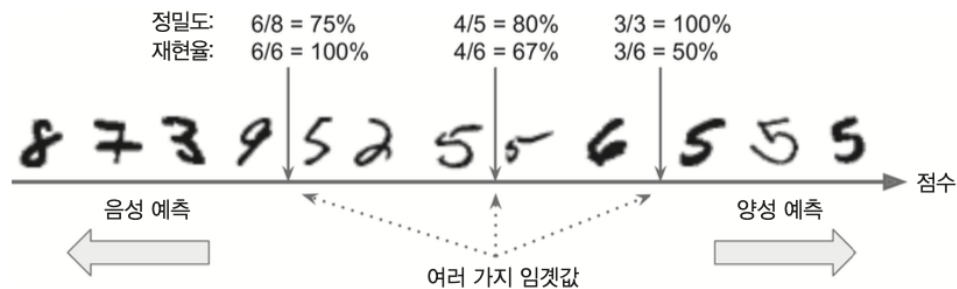
- F_1 점수가 높을 수록 분류기의 성능을 좋게 평가하지만, 경우에 따라 조심할 필요가 있음
 - 경우에 따라 재현율과 정밀도 둘 중의 하나에 높은 가중치를 두어야 할 때가 있기 때문
- 앞서 정의된 F_1 점수는 재현율과 정밀도의 중요도가 동일하다고 가정한 결과
- 예를 들어, 암 진단 결과와 관련해서 아래 두 경우에 발생하는 비용이 다름
 - 실제로 음성지만 양성으로 오진(정밀도)
 - 실제로 양성이지만 음성으로 오진(재현율)

정밀도/재현율 트레이드오프

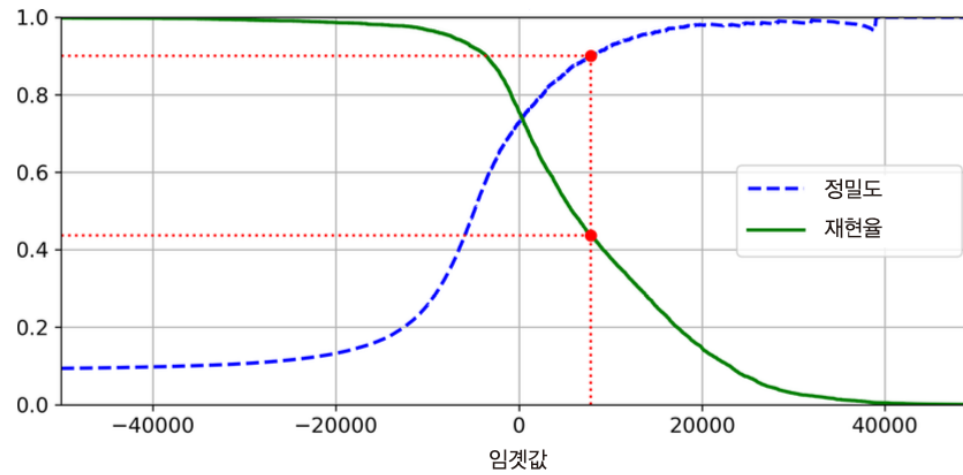
- 정밀도와 재현율 상호 반비례 관계
- 정밀도와 재현율 사이의 적절한 비율 찾기

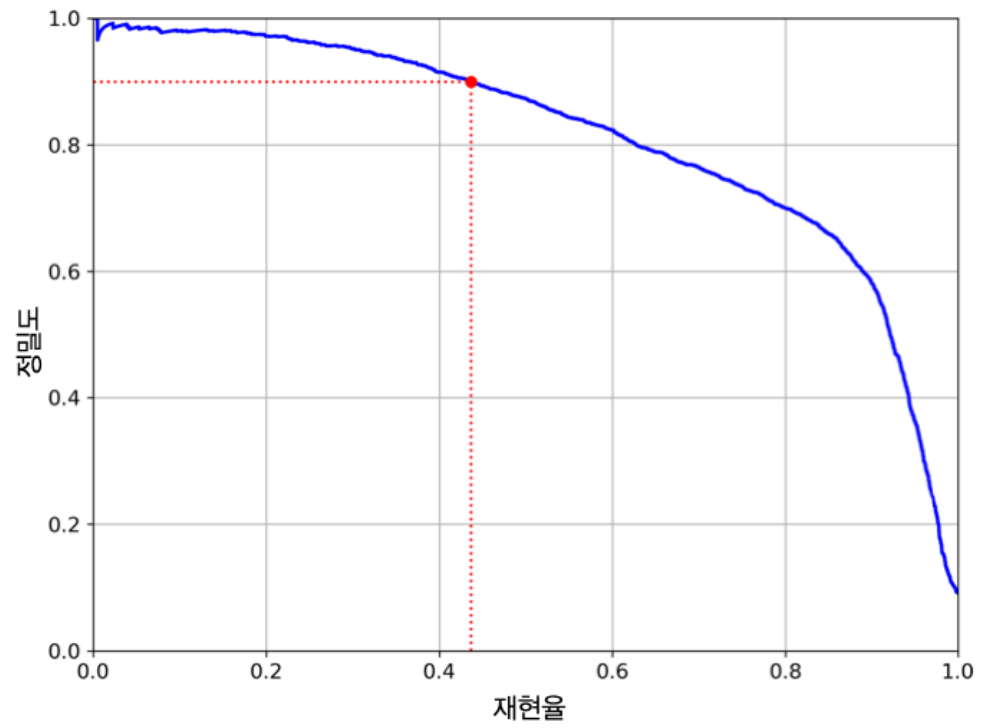
결정 함수와 결정 임계값

- 결정 함수(decision function): 분류기가 각 샘플의 점수를 계산할 때 사용
- 결정 임계값(decision threshold): 결정 함수의 값이 이 값보다 작거나 크면 양성 클래스로 분류, 아니면 음성 클래스



- 임계값 커질 수록
 - 정밀도 올라감
 - 재현율 떨어짐





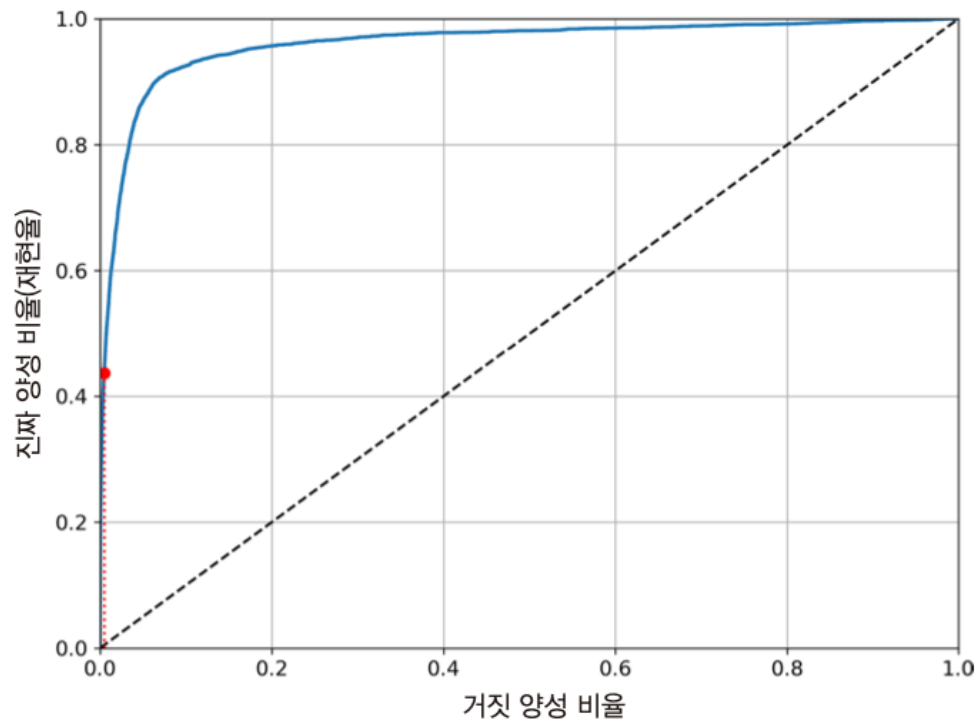
ROC 곡선과 AUC

- 수신기 조작 특성(receiver operating characteristic, ROC) 곡선을 활용하여 이진 분류기의 성능 측정 가능

ROC 곡선

- 결정 임계값에 따른 거짓 양성 비율(false positive rate, FPR)에 대한 참 양성 비율(true positive rate, TPR)의 관계를 나타낸 곡선
- 참 양성 비율: 재현율
- 거짓 양성 비율: 원래 음성인 샘플 중에서 양성이라고 잘못 분류된 샘플들의 비율
 - 예를 들어, 손글씨 숫자 분류기가 결정 임계값이 0일 때 정해진 거짓 양성 비율은 아래와 같음

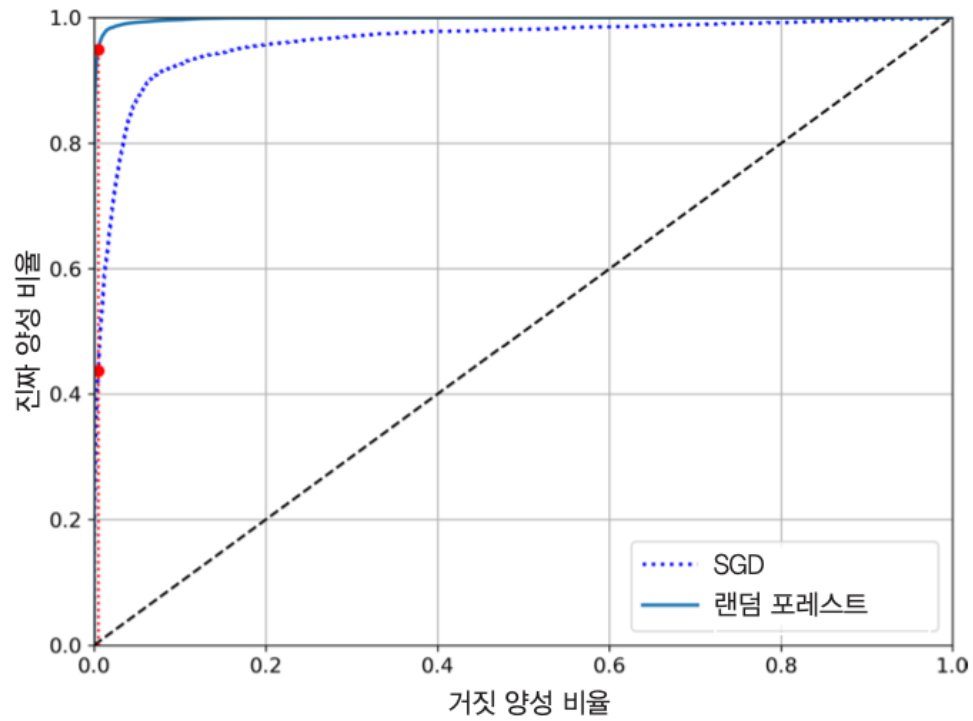
$$FPR = \frac{FP}{FP + TN}$$



AUC와 분류기 성능

- 재현율(TPR)과 거짓 양성 비율(FPR) 사이에도 서로 상쇄하는 기능이 있다는 것을 확인 가능
 - 재현율(TPR)을 높이려고 하면 거짓 양성 비율(FPR)도 함께 증가
- 따라서 좋은 분류기는 재현율은 높으면서 거짓 양성 비율은 최대한 낮게 유지해야함
- ROC 곡선이 y축에 최대한 근접하는 결과가 나오도록 해야함.
- AUC: ROC 곡선 아래의 면적
 - 이 면적이 1에 가까울 수록 성능이 좋은 분류기로 평가

- SGD와 랜덤 포레스트의 AUC 비교



3.4 다중 클래스 분류

다중 클래스 분류기(multiclass classifier)

- 세 개 이상의 클래스로 샘플을 분류하는 예측기
- 다항 분류기(multinomial classifier)라고도 부름
- 예를 들어, 손글씨 숫자 분류의 경우 0부터 9까지 10개의 클래스로 분류해야 함

다중 클래스 분류 지원 분류기

- SGD 분류기
- 랜덤 포레스트 분류기
- 나이브 베이즈(naive Bayes) 분류기

이진 분류만 지원하는 분류기

- 로지스틱 회귀
- 서포트 벡터 머신

이진 분류기 활용

- 이진 분류기를 활용하여 다중 클래스 분류 가능
 - 일대다(OvR 또는 OvA)
 - 일대일(OvO)

일대다 방식(OvA 또는 OvR)

- OvA(One-versus-All) 또는 OvR(One-versus-Rest)
- 숫자 5 예측하기에서 사용했던 이진 분류 방식을 동일하게 모든 숫자에 대해서 실행
- 각 샘플에 대해 총 10번 각기 다른 이진 분류기를 실행
- 이후 각 분류기의 결정 점수 중에서 가장 높은 점수를 받은 클래스를 선택

일대일 방식 (0v0)

- One-versus-One
- 조합 가능한 모든 일대일 분류 방식을 진행하여 가장 많은 결투(duell)를 이긴 숫자를 선택
- MNIST의 경우, 아래와 같이 총 $9+8+\dots+1 = 45$ 개의 결투를 판별하는 분류기를 이용
 - 훈련 세트의 각 샘플에 대해 총 45번 결투가 벌어지며 그중에서 가장 높은 점수를 얻는 숫자 k 가 선택됨

예제: OvO 활용

- 서포트 벡터 머신(SVC)
 - OvO 선호
- 대부분의 이진 분류기는 일대다 전략을 선호

예제: OvR 활용

- 이진 분류기를 일대일 전략 또는 일대다 전략으로 지정해서 학습하도록 만들 수 있음.
- 사이킷런의 경우: `OneVsOneClassifier` 또는 `OneVsRestClassifier` 사용

다중 클래스 지원 분류기

- SGD 분류기는 다중 클래스 분류를 직접 지원
- 따라서 사이킷런의 OvR, OvO 등을 적용할 필요 없음

다중 클래스 분류기 성능 측정

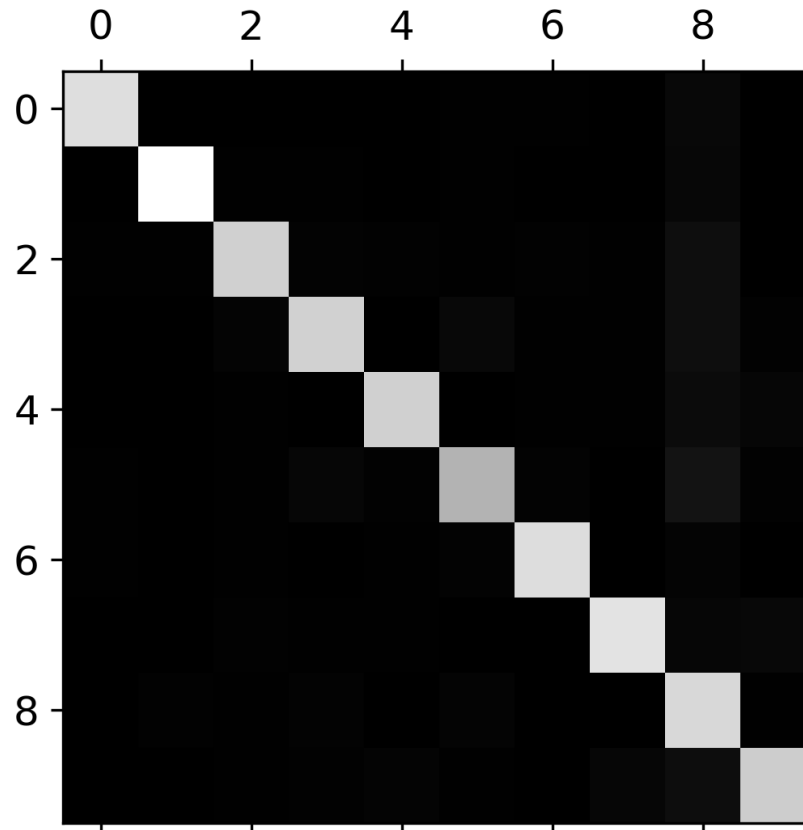
- 다중 클래스 분류기의 성능 평가는 교차검증을 이용하여 정확도를 측정
- MNIST의 경우 0부터 9까지 숫자가 균형 있게 분포되어 있어서 데이터 불균형의 문제가 발생하지 않음.

3.5 에러 분석

- 가능성이 높은 모델을 하나 찾았다고 가정하고 이 모델의 성능을 향상시킬 방법 모색
- 예제: 에러 종류 분석하기

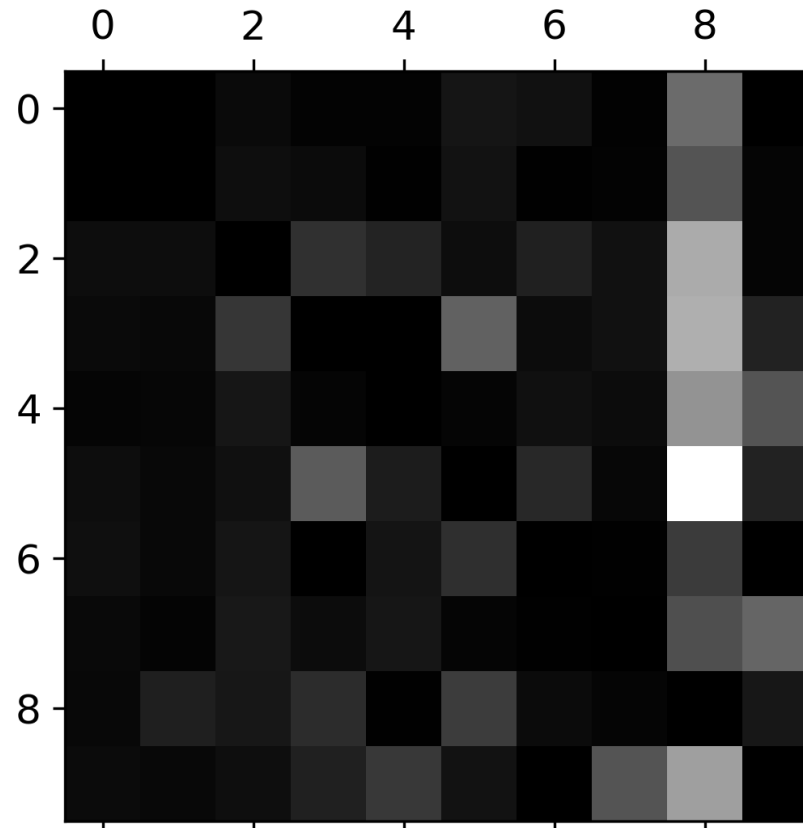
오차 행렬 활용

- 손글씨 클래스 분류 모델의 오차 행렬을 이미지로 표현 가능
- 대체로 잘 분류됨: 대각선이 밝음.
- 5행은 좀 어두움. 숫자 5의 분류 정확도가 상대적으로 낮음

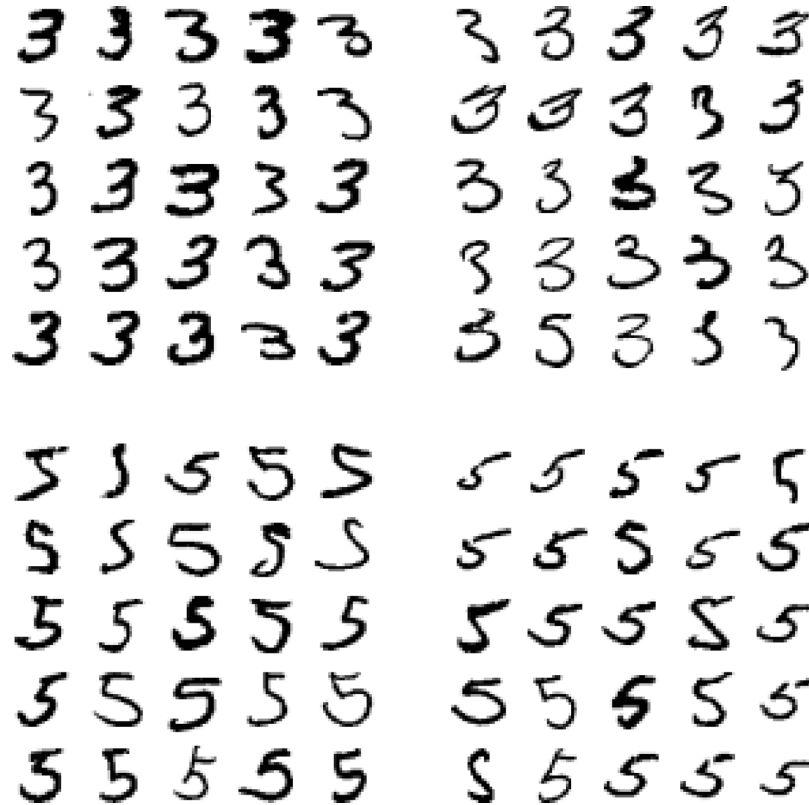


오차율 이미지

- 8행이 전반적으로 어두움.
 - 8은 잘 분류되었다는 의미임.
- (3, 5)와 (5,3)의 위치가 상대적으로 밝음.
 - 3과 5가 서로 많이 혼동됨.



- 3과 5의 오차행렬 그려보기
 - 음성: 3으로 판정
 - 양성: 5로 판정



- 3과 5의 구분이 어려운 이유
 - 선형 모델인 SGD 분류기를 사용했기 때문
 - 픽셀마다 가중치를 적용하여 단순히 픽셀 강도에만 의존하기 때문
- 이미지 분류기의 한계
 - 이미지의 위치나 회전 방향에 민감함
 - 이미지를 중앙에 위치시키고 회전되지 않도록 전처리하면 좀 더 좋은 성능 가능

3.6 다중 레이블 분류

- 샘플마다 여러 개의 클래스 출력

예제: 얼굴 인식 분류기

- 한 사진에 여러 사람이 포함된 경우, 인식된 사람마다 하나씩 꼬리표(tag)를 붙여야 함.
- 앨리스, 밥, 찰리의 포함여부를 확인 할 때, 밥이 없는 경우:
 - [True, False, True] 출력

예제: 숫자 분류

- 7 이상인지 여부와 함께 홀수 여부도 동시 출력
- 5가 입력될 때:
 - [False, True] 출력

다중 레이블 분류 지원 모델

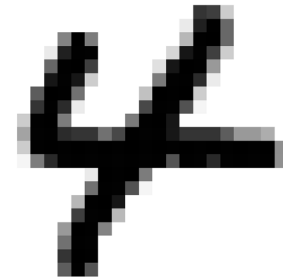
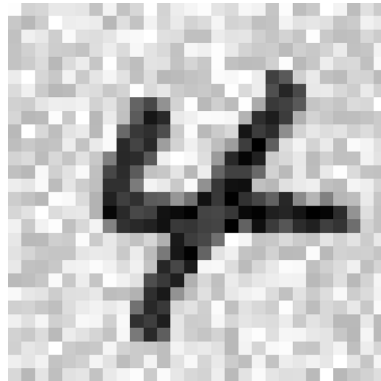
- k-최근접 이웃 분류기
- 사이킷런의 `KNeighborsClassifier`
- 다중 레이블 분류기를 평가하는 방법은 다양함
 - 모든 레이블의 가중치가 같다고 가정: 각 레이블의 F1 점수를 구하고 평균 점수를 계산
 - 가중치: 레이블에 클래스의 지지도(즉, 타깃 레이블에 속한 샘플 수)를 가중치로 사용 가능

3.7 다중 출력 분류

- 원래 다중 출력 다중 클래스 분류라고 불림
- 다중 레이블 분류에서 한 레이블이 다중 클래스가 될 수 있도록 일반화한 것
- 다중 레이블 분류기 지원 모델: k-최근접 이웃 분류기
 - 사이킷런의 `KNeighborsClassifier`

예제: 이미지에서 잡음을 제거하는 시스템

- 잡음이 많은 숫자 이미지를 입력으로 받고, 깨끗한 숫자 이미지를 MNIST 이미지처럼 픽셀의 강도를 담은 배열로 출력
- 아래 왼쪽: 잡음 포함된 사진
- 아래 오른쪽: 깨끗한 타겟 이미지



- 다중 레이블: 각각의 픽셀이 레이블 역할 수행
- 다중 클래스: 레이블이 0부터 255까지 픽셀 강도를 가짐.
- 분류기를 훈련시켜 잡음이 섞인 입력 이미지를 깨끗한 타깃 이미지로 만들어 보기
- 아래 사진: 분류기가 만들어낸 이미지

