4. 특성공학

- 특성공학(feature engineering)
  - 특정 애플리케이션에 가장 적합한 데이터 표현을 찾는 기법.
  - 매개변수 튜닝에 앞서 기본적으로 진행되어야 할 작업.

- 연속형 특성(continuous feature)
  - 실수형 데이터 형식.
  - 꽃잎의 길이, 너비, 나이, 무게 등.

- 범주형 특성(categorical features, discrete features)
  - 숫자가 아님. 중간값, 순서 없음.
  - 제품의 브랜드, 색상 등.

4. 문자열 변수의 one-hot encoding

- 예제 : 개인의 특징을 기반으로 수입이 5만불 이상인지 여부를 예측하는 문제(분류)
  - 연속형 특성 예 : 나이, 주당 근무시간...
  - 범주형 특성 : workclass, education, sex, occupation...

- (실습) one-hot encoding 방법
  - 개별 특성에 나타나는 모든 종류의 값에 대하여 0,1로만 표현하는 방법
  - 하나의 값만 1로 표현되고 나머지는 0으로 표현됨.
  - workclass 에 있는 값의 종류 : Government Employee, Private Employee, Self Employed, Self Employed Incorporated,
  - workclass.unique(), workclass.value_counts()로 확인.

workclass	Government Employee	Private Employee	Self Employed	Self Employed Incorporated
Government Employee	1	0	0	0
Private Employee	0	1	0	0
Self Employed	0	0	1	0
Self Employed Incorporated	0	0	0	1

4. 숫자표시 변수의 경우

- 특성값이 숫자로 되어있지만 범주형 변수인 경우도 있음
- workclass 원본데이터에서 문자열 대신 다음과 같이 표현했을 경우,
- Government Employee = 1, Private Employee = 2, Self Employed = 3, Self Employed Incorporated = 4
- 이 경우는 연속형 변수로 다루면 안됨.
- 이러한 숫자특성 변수를 one-hot encoding으로 변환하려면
- pd.get_dummies()에 특성 이름을 명시해야 함.
- (실습)

4. ColumnTransformer()

- Scikit-learn의 ColumnTransformer 이용
- OneHotEncoder() 기본값은 연속형 데이터를 구별하지 않고 변환함.
- OneHotEncoder()를 이용하여 필요한 열만 밤주형 변수변환을 수행할 수 있음.
- ColumnTransformer() 함수에서는 범주형 특성 변환 뿐 아니라 데이터 scale 조정도 수행할 수 있음.
- (실습)

4. 다항식 확장

- 연속형 특성을 확장하는 법중 하나
  - 특성 x가 주어지면 x**2, x**3, x**4... 시도 가능.

- mglearn.make_wave() 데이타셋에 선형회귀를 적용.
  - 다항식 특성을 추가하고 원본 데이터에 선형회귀식을 적용하면 부드러운 비선형 예측곡선 생성 가능.
  - SVR을 적용했을 때 형성되는 비선형 예측곡선과 유사한 예측곡선을 생성함.
  - (실습)

- 보스턴 주택데이타셋에 적용
  - 원본은 13개의 특성이 존재함.
  - 2차 다항식을 적용하면 특성이 105개로 늘어남.
  - 상호작용 특성이 없을 때 / 있을 때 테스트 점수 = 0.621 / 0.753
  - (실습)

4. 비선형 변환(log, exp)

- 특성과 목표값 사이에 비선형성이 있는 경우 선형회귀 모델은 성능이 저하됨.
  - 확률 요소를 가진 많은 알고리즘 들은 정규분포를 가정하고 있음(정규분포일 때 최고의 성능)
  - 원본 데이터에 변형을 가해 원본 데이터(특성, 목표값 모두)을 정규분포로 만드는 것이 바람직함.

- 예제 : 사용자가 얼마나 자주 로그인하는가?
  - 이러한 데이터는 포아송 분포를 따름.
  - (포아송 분포 : 단위시간당 사건이 일어나는 횟수의 확률분포)
  - log(포아송 분포값) → 정규분포 로 변환
  - 원본데이타 / log 변환 데이터의 테스트 점수 = 0.622 / 0.875
  - (실습)

- 위의 예에서는 원본의 3 특성이 모두 포아송 분포였음.
- 실제 상황에서는 속성마다 다른 분포를 가지는 경우가 대부분 → 특성마다 적절한 변환 적용 필요.

- (특성값이 아닌) 목표값에 변형 적용필요한 경우도 있음.
  - 예) 목표값 : 단위시간당 주문횟수 예측
  - 주) 트리기반 모델에서는 이러한 변환이 필요없음.

4. 특성자동선택 – 일변량 통계기반 특성자동선택

- 고차원 데이타셋을 다룰 때 유용한 특성만 선택해야 모델이 단순해지고 일반화 성능이 올라감.
- 각 특성과 목표값 사이의 통계적 관계를 계산
- 계산값이 큰 순서대로 특성 선택
- 각 특성이 독립적으로 평가됨(특성간의 상호작용에 의한 영향은 고려되지 않음)
- 특성 선택 후 적용하는 모델은 제한이 없음.

- (실습)
- 암 데이타셋에 50개의 잡음특성을 추가하고 선택성능 확인.
  - 원본 특성만으로 훈련시 테스트 점수 : 0.951
  - 50개의 잡음특성 추가 후 40개 잡음 제거 후 테스트 점수 : 0.933

4. 특성자동선택 – 모델기반 특성선택

- 본 학습모델이 사용할 특성을 선택하기 위해 다른 지도학습 모델을 사용.
- 모든 특성을 상호작용을 반영(일변량 분석은 특성을 독집적으로 평가)
- SelectFromModel() 함수에서 지정한 임계치보다 큰 모든 특성을 반환함
- (실습)
  - 암데이타셋에 대해 RandomForest 알고리즘으로 특성을 선택하고 LogisticRegression 알고리즘으로 분류 실행.

4. 특성자동선택 – 반복적 특성선택

- 모든 특성조합을 대상으로 최적의 특성조합을 탐색하는 것은 큰 비용이 듬(2^N 개의 조합).
- 좀 더 효율적인 방법으로 최적의 특성을 선택하여 모델에 적용할 목적.
- 전진방법
  - 특성선택이 0인 상태에서 모든 특성을 대상으로 하나씩 순차적으로 선택.평가하여 가장 높은 성능보이는 특성 선택.
  - 위 결과를 기반으로 두번 째 특성을 선택.
  - 다중공선성 문제로 인하여 반복 탐색시 특성의 중요도는 이전 선택의 영향을 받아 변동됨.
  - 지정한 개수의 특성이 선택되거나 성능의 개선이 없을 때까지 반복작업

- 후진 방법
  - 모든 특성을 대상으로 하나씩 순차적으로 제거하면서 성능을 평가
  - 가장 성능이 안나오는 특성을 제거.
  - 위 결과를 기반으로 두번째 특성을 평가하여 제거.
  - 지정한 개수의 특성이 선택되거나 성능의 개선이 없을 때까지 반복작업

- (실습)
  - 잡음이 포함된 암데이터셋(80개 특성)에 RFE(Recursive Feature Elimination) 적용
  - 원본 30개 특성 중 29개 선택됨, 기타 11개 특성 추가선택(실행시 지정)

- 특성 선택/제거시마다 모델을 구축해야하므로 비용이 많이 듬.

4. 특성선택 – 전문가 지식활용

- 데이터를 표현할 때 특성의 내용에 따라서 전문지식을 적용해야 할 경우.
- 예제) 자전거 대여소의 일자별, 시간대별 대여횟수 예측.
  - 데이터 : 8.1 ~ 8.31 기간, 3시간 간격의 대여횟수
  - 컴퓨터 시스템에서 시간정보의 저장은 POSIX 표현(연속형 정수)으로 저장됨.
  - 2015-08-01 00:00:00 = 1438387200000000000
  - 이는 시간, 요일 등의 정보를 표현하지 못하므로 학습시 문제가 됨.

- (실습)
- 원본데이타를 RandomForestRegressior로 예측 :
  - R^2 : -0.04
  - 특성값이 날짜 정보 없이 연속된 숫자이므로 숫자로 표현된 미래는 예측불가
  - 특성값(X축)을 POSIX 표현에서 시간으로 변경표현 : R^2 → 0.60
  - 특성값(X축)을 시간표현에서 요일까지 표현 : R^2 → 0.84

원본데이타를 LinearRegression로 예측 :

- R^2 : 0.13 ← 요일, 시간이 연속형 변수로 해석되기 때문.
- 요일, 시간을 범주형으로 변경할 경우(요일 : 7개 범주, 시간 : 8개 범주) : R^2 → 0.62
- onehot 표현에 다항식 확장을 적용했을 경우 : : R^2 → 0.85

5. 모델 성능평가 – 기존의 모델개발과정

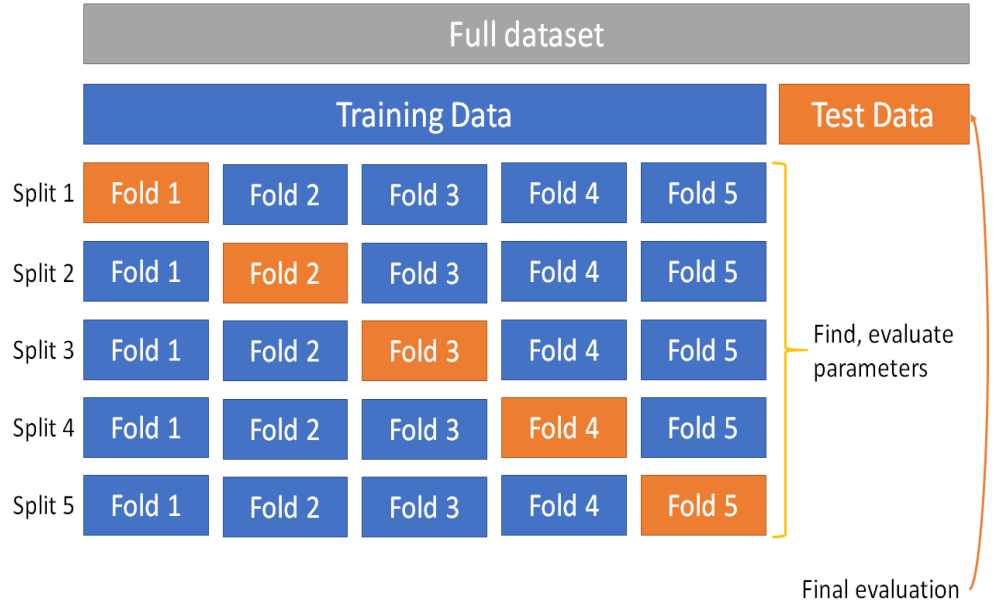
- 일반적인 모델 개발과정
  - 전체 데이터셋을 훈련데이터셋 + 테스트 데이터셋으로 나눔
  - 테스트셋을 나누는 이유는 생성된 모델이 새로운 데이터를 예측하는 성능을 확인하기 위함)
  - fit(Train set) 명령으로 훈련세트를 대상으로 모델을 만듬.
  - score(Test set) 명령으로 테스트 세트를 대상으로 모델의 성능을 평가.

- (기존 모델 구축과정에서)
- 평가의 척도는 분류는 정확도, 회귀는 R^2 사용.

5. 모델 성능평가 - 교차검증

- 목적
  - 모델의 일반화 성능 신뢰성 향상
- 방법
  - 데이타셋을 여러 번 반복해서 나누고 여러 모델을 학습함.
  - (5-겹 교차검증의 경우)
  - 다섯 개의 묶음으로 분할
  - 첫번째 모델은 2 ~ 5번째 묶음으로 훈련하고 1번째 묶음으로 성능을 측정
  - 묶음을 바꿔가며 5번 반복 시행
  - 다섯개 모델의 성능측정값의 평균으로 모델의 성능을 정의함.
- (실습)
  - 붓꽃 데이터셋에 대해 5겹/10겹 교차검증

교차검증 모형

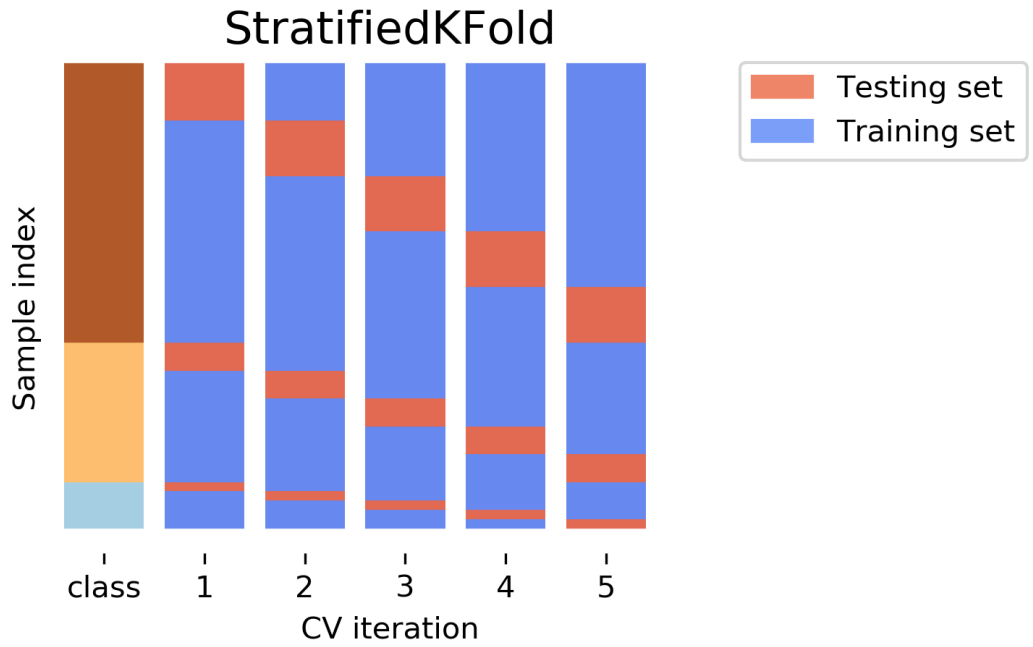


- 교차검증의 장단점
  - 기존 방법은 test 데이타가 편중되어 검증결과가 왜곡될 수 있음.
  - 교차검증은 모든 데이터가 한번씩 검증데이타에 포함되므로 더 안정적인 결과를 얻을 수 있음.
  - 분할을 한번 했을 때보다 테스트 데이터 양이 늘어남.
  - 75:25로 한번 분할하면 훈련데이타로 75%만 사용가능하지만, 5-fold를 적용하면 80%, 10-fold를 적용하면 90% 데이터를 훈련에 사용할 수 있음.
  - 단점 : 폴드만큼 계산량 증가

5. 모델 성능평가 – 계층별 교차검증

- 계층별 (stratified k-fold) 교차검증
- 분류 문제에 있어서 조건없이 폴드를 구성하면 클래스가 편중될 수 있음.
  - ➢ 폴드 안의 클래스 비율이 전체 데이타셋의 클래스 비율과 같도록 폴드를 생성.

- (실습)



5. 모델 성능평가 - 그룹별교차검증

- 그룹별(GroupKFold) 교차검증
- 예) 얼굴사진에서 표정을 판단하는 데이타셋 구성
  - 한 사람의 데이터는 여러가지 표정의 여러 개 데이터로 구성됨.
  - 특정인의 데이터는 훈련세트와 테스트세트에 동시에 분포하면 테스트세트에 훈련세트의 정보가 누출됨.
  - 훈련세트와 테스트세트는 서로 다른 사람의 데이터로 구성되어야 함.
  - GroupKFold 적용

- (실습)

5. 모델 성능평가 – 그리드 서치

- 그리드서치
  - 매개변수를 튜닝하여 일반화성능 개선하는 방법
  - 모델에 따라 중요한 매개변수의 가능한 조합을 대상으로 성능 평가.

- 예제 : 붓꽃 분류 예제에 SVC를 적용하여 예측모델을 개발
  - 중요 매개변수인 gamma, C를 대상으로 최적의 매개변수를 찾아야 함.
  - 다음과 같이 gamma, C 매개변수를 적용한다면 총 36개 조합을 시험해야 함.
  - gamma : 0.001, 0.01, 0.1, 1, 10, 100
  - C : 0.001, 0.01, 0.1, 1, 10, 100

- (실습)
- 데이터셋을 훈련세트와 테스트셋으로 나누고 그리드 서치 실행
- 결과
  - 최적 파라미터: { ' C ' : 100, ' gamma ' : 0.001}
  - 최고 점수: 0.97

- 가장 성능이 높게 나오는 매개변수 조합 선택
- → 0.97 이라는 모델의 성능을 신뢰할 수 있는가?

5. 성능평가 – 기존 방법의 문제점

- 훈련셋, 테스트셋으로 최상의 모델 생성
  - Training set은 모델을 학습하는데 사용.
  - test set은 training set으로 만들어진 모델의 성능을 측정하기 위해 사용.
  - 일반적으로 어떤 모델이 가장 데이터에 적합한지 찾아내기 위해서 다양한 파라미터와 모델을 사용해보게 되며, 그 중 validation set으로 가장 성능이 좋았던 모델을 선택(cross validation 무관).

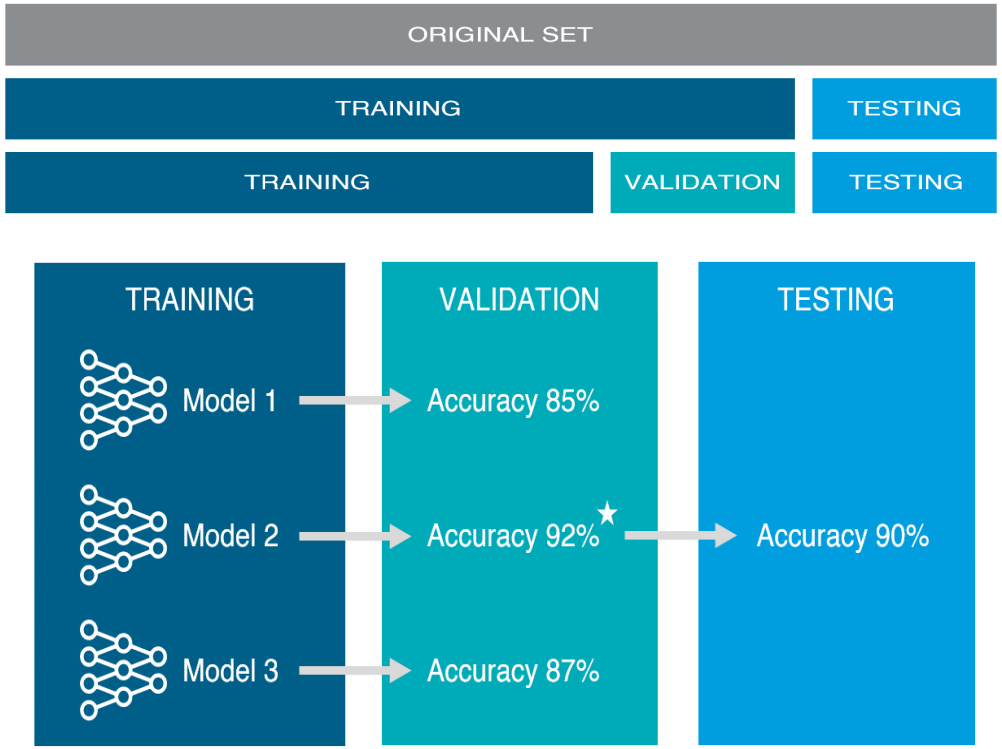
- 훈련셋-검증셋의 문제점
  - 최고성능이 나온 원인이 최적의 매개변수 조합 때문일 수도 있고,
  - **데이터셋이 최적 매개변수 조합과 잘 맞는 데이터 집합이어서 일수도 있음.**

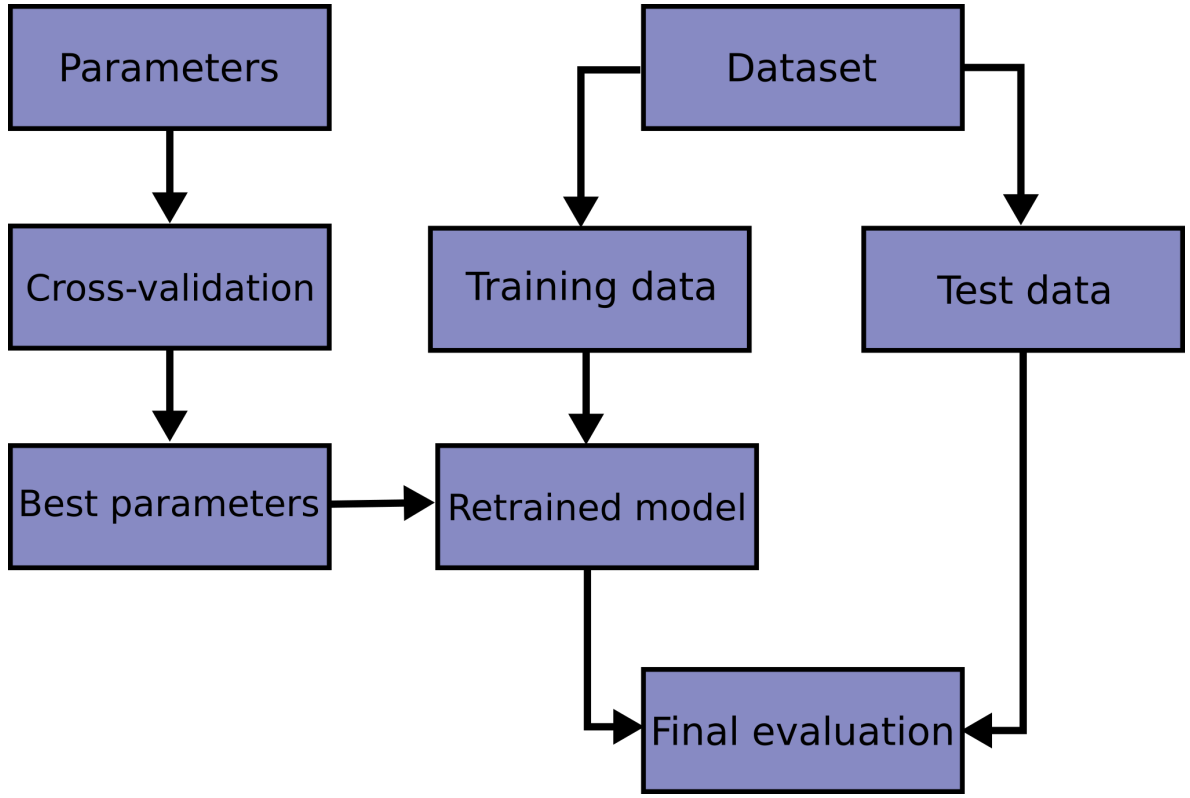
5. 모델 성능평가 – 개선된 모델 평가방법

- 데이터셋을 훈련셋, 검증셋, 테스트셋으로 나눔
- 훈련셋, 검증셋으로 그리드 서치 실행하여 최적 매개변수 찾음.
- 찾은 최적 매개변수로 학습된 모델로 테스트셋을 대상으로 최종 성능 평가
- 테스트 셋 사용
  - 마지막으로 딱 한번 해당 모델의 예상되는 성능을 측정하기 위해 사용
  - 앞으로 기대되는 성능을 예측하기 위해 사용
- (실습)
  - 검증 세트에서 최고 점수: 0.96
  - 최적 파라미터: {'C': 10, 'gamma': 0.001}
  - 최적 파라미터에서 테스트 세트 점수: 0.92
  - **→ 최적 매개변수와 테스트점수가 변경됨.**

- 최종모델 완성
  - 기존 training set만을 사용하였던 모델의 파라미터와 구조는 그대로 사용.
  - 전체 데이터를 사용하여 다시 학습시킴으로써 모델개선.

5. 모델 성능평가 – 교차검증과 그리드 서치를 통합한 모델구축 흐름





5. 모델 성능평가 – GridSearchCV()

- 기본 그리드서치의 문제점
  - 데이터가 부족할 경우 모델이 불안정할 수 있음.
- GridSearchCV()
  - 모델성능을 더 신뢰성있게 구축하기 위해서는 그리드서치와 교차검증을 통합할 필요 있음.
  - 이전 예제에 5겹 교차검증을 적용할 경우 36 * 5 = 180 번의 모델생성이 필요함.
  - 각 매개변수 조합성능 측정시 교차검증 방법으로 시행.
  - 매개변수 설정마다 5번 교차검증분할, 5개의 검증성능 산출 → 평균 계산.
- (실습)
  - 붓꽃데이터셋에 대해 SVC(매개변수 : C, gamma), cv=5 조건으로 GridSearchCV 수행.

- grid_search.fit(X_train, y_train)
  - param_grid에 설정된 매개변수 조합에 대해 교차검증 수행.
  - 최적의 매개변수 탐색
  - 최적 매개변수로 전체 훈련데이터셋을 대상으로 새로운 모델 생성 → best_estimator_
- grid_search.best_score_
  - - 특정 매개변수 설정으로 각 분할에서 얻은 교차검증 정확도의 평균
- grid_search.score(X_test, y_test)
  - 생성된 최적의 모델로 테스트셋을 대상으로 평가한 결과.

5. 모델 성능평가 – 교차 검증 결과 분석

- 히트맵으로 매개변수 조합에 따른 성능의 변화를 볼 수 있음.
- 결과에 따라 매개변수의 범위를 조정하여 최적의 매개변수 찾을 수 있음.
- (실습)
  - C, gamma 매개변수의 변화에 따라 0.37 ~ 0.97까지 성능의 차이가 비교적 큼(민감).
  - C의 윗쪽, gamma의 왼쪽으로 성능의 개선 여지가 있음.

- (실습2 - 부적절한 매개변수 범위 설정 분석)
  - 첫번째 사례(모두 같은 성능)

    → 매개변수 설정이 잘못되거나, 영향이 없는 매개변수일 경우
  - 두번째 사례(세로 띠 형태)

    → 가로 매개변수는 적절하나 세로 매개변수는 영향을 미치지 못함(범위 부적절 또는 무영향 매개변수)
  - 세번째 사례(고성능 매개변수 분포가 치우침)

    → 고성능 매개변수 지역의 너머에 더 좋은 매개변수가 있을 가능성이 있음.

- 최종 테스트셋을 대상으로 매개변수를 평가해서는 안됨.

5. 모델 성능평가 – 이진분류 평가지표

- 지금까지 사용한 성능평가지표
  - 분류 : 정확도, 회귀 : R^2

- 불균형 데이타셋
  - 두 클래스 중 한쪽이 월등하게 많은 경우
  - 매우 흔한 상황임(예: 신용카드 부정사용 데이터, 품질불량 데이터 등)

- 이진분류 에러의 종류
  - 양성 : 일반적으로 두 클래스 중 빈도가 낮은 클래스
  - 음성 : 두 클래스 중 빈도가 높은 클래스
  - 거짓양성 : 음성클래스를 양성을 판정하는 경우(예: 정상환자를 암환자로 판정)
  - 거짓음성 : 양성클래스를 음성을 판정하는 경우(예: 암환자를 정상환자로 판정)

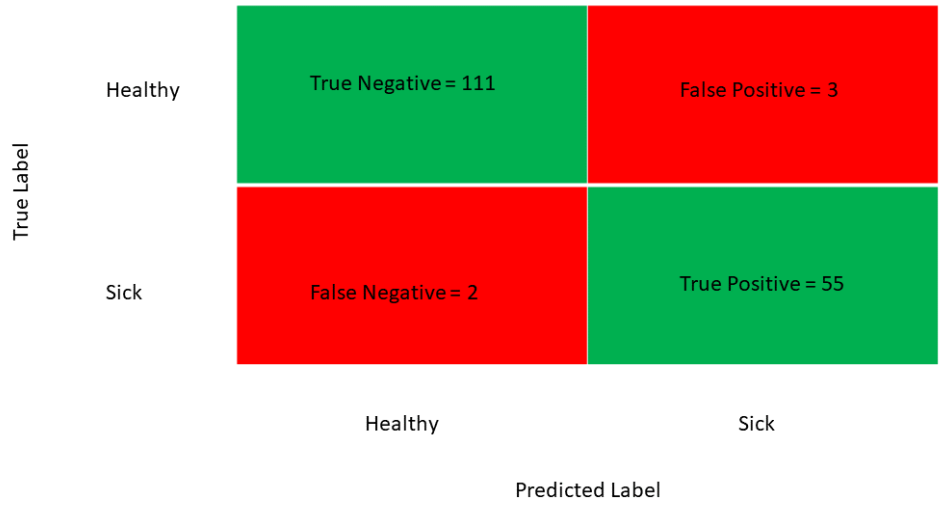
- 정확도(accuracy)로 다양한 알고리즘 성능 평가
  - 정확도 : 정확하게 분류한 양성,음성 샘플 수 / 전체샘플수
  - 정확도는 불균형 데이터셋에 적용하면 분류성능을 적절하게 알려주지 않음.

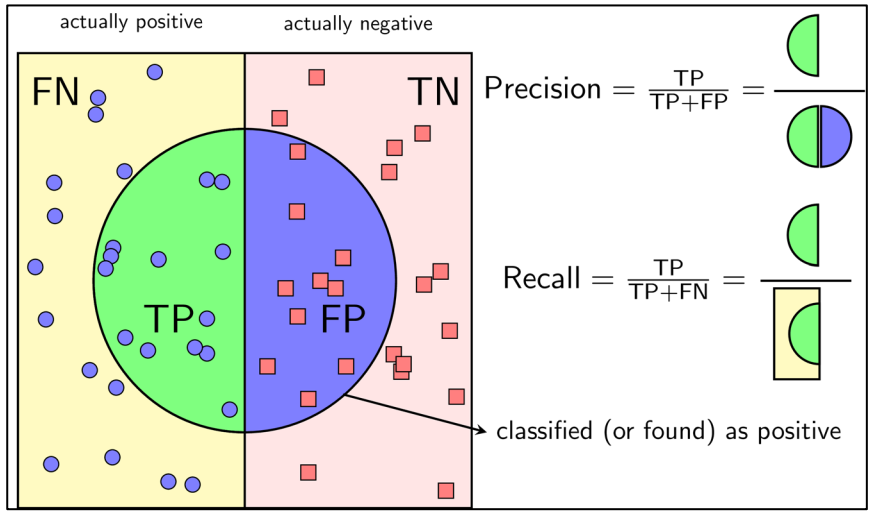
- (실습 : 불균형 데이터셋 - 정확도로 평가)

5. 모델 성능평가 – 오차행렬(confusion matrix)

- 분류문제 예측의 결과를 요약해서 보여주는 표.
- 정밀도(precision) : 정상 양성예측 샘플 수 / 전체 **양성예측** 샘플 수
  - 양성예측 결과중 양성만을 정확하게 예측하는 척도,
  - 거짓 양성(음성을 양성으로 예측)을 줄이는 것이 목표일 때. 예: 신약의 치료효과
- 재현율(recall, 또는 민감도:sensitivity) : 정상 양성예측 샘플 수 / **전체 양성** 샘플 수
  - 실제 양성 샘플 중 얼마나 정확하게 분류했는지 측정.
  - 거짓음성(양성을 음성으로 예측)을 줄이는 것이 목표일 때. 예: 암환자 감지
- f-점수(f-score) : 정밀도와 재현율의 조화평균
  - 종합 성능평가용

$$\text{F1-score} = 2.\frac{\text{Precision}.\,\text{Recall}}{\text{Precision} + \text{Recall}}$$





5. 불확실성 고려 - 임계값 조정

- 대부분의 분류기는 예측의 확신을 정량화하는 함수를 제공
- decision_function() : 예측확신을 -∞ ~ ∞ 사이의 값으로 표현. 기본 임계값 = 0
- predict_proba() : 예측확신을 0 ~ 1.0 사이의 값으로 표현. 기본 임계값 = 0.5
- 필요에 따라 정밀도/재현율을 개선할 필요가 있을 때 임계값 조정
- 예제) 재현율을 높이고자 할 경우(TP늘이는 방향)
  - 암진단의 경우 거짓양성이 늘어나더라도 진짜 양성을 놓치면 안되는 경우
  - 대신 거짓양성도 늘어나게 됨.

(실습)

  - 임계값을 바꾸어 양성으로 분류가 늘어나도록 조정 : 0 → -0.8

5. 정밀도-재현율(precision-recall curve) 곡선

- 모델의 목적에 따라 임계값을 바꾸는 것은 모델의 정밀도-재현율의 상충관계를 조정하는 작업.
- 임계값을 조정하여 재현율 또는 정밀도 목표를 맞추면서, 다른 지표를 높이는 임계값을 찾는 것이 중요함.
- 모든 임계값에 따라 정밀도-재현율을 평가할 수 있는 곡선 이용하여 최적 임계값 판단.
- 서로 다른 모델의 성능을 비교하는데도 사용 가능.
- (실습 : 정밀도-재현율 곡선)

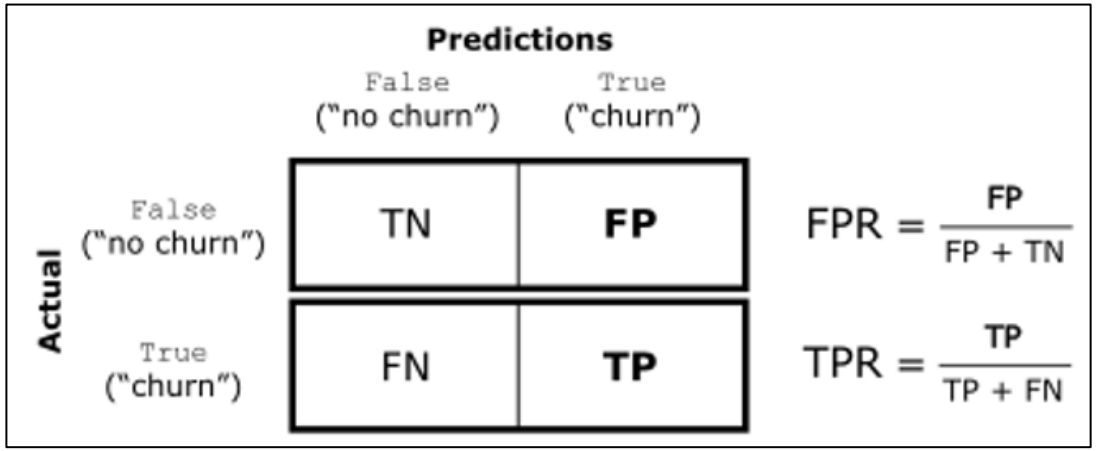
- 해석
  - 곡선이 오른쪽 위 지점의 임계값이 재현율, 정밀도 값이 좋아지는 방향임.
  - 오른쪽으로 이동하면 정밀도는 높아지지만 재현율은 낮아지는 임계값임.

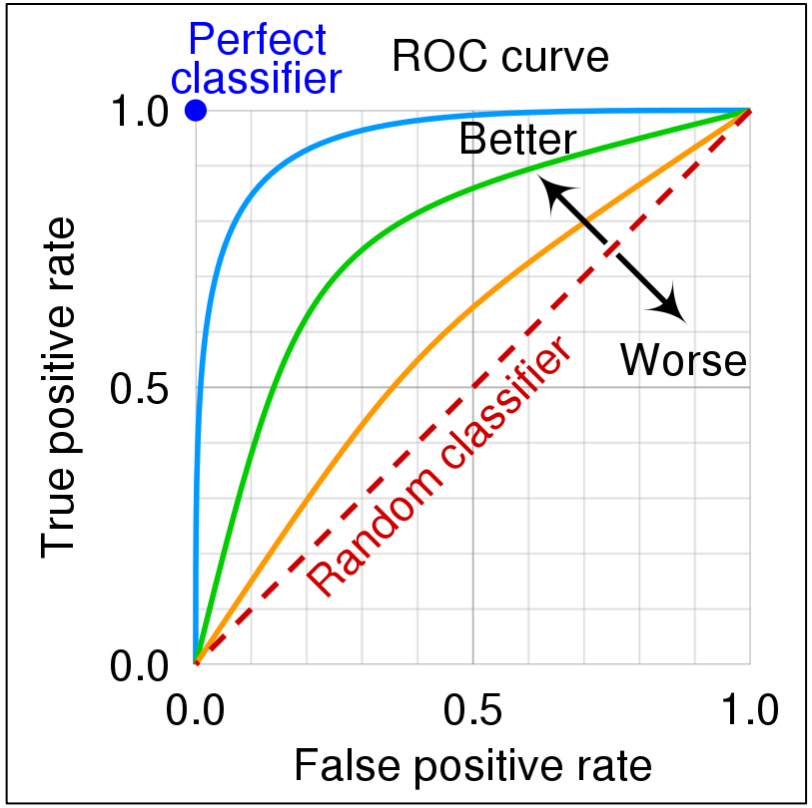
- 두 모델의 성능을 비교할 수도 있음.
- 그래프가 아닌 하나의 숫자로 성능을 파악하고자 할 경우 평균 정밀도(avarage_precision) 값 사용.
  - 평균정밀도 : 재현율-정밀도 곡선의 면적

- (실습 - 평균정밀도 값)

5. 모델 성능평가 – ROC, AUC

- ROC(Receiver Operating Characteristic) 곡선
  - 모든 임계값에 대하여 정밀도와 재현율 대신 진짜양성비율(TPR)에 대한 거짓양성비율(FPR) 값을 그린 곡선.
  - 불균형데이타셋의 경우도 무작위예측값은 0.5(클래스분포에 영향을 받지않음)이므로 정확도 왜곡이 없음.
  - 왼쪽위(낮은 FRP값, 높은 TPR값) 지점이 좋은 성능 임계값임.
  - 그래프 모양을 보고 최적의 임계값 지점 선택 가능.
- AUC(Area under the Curve)
  - 곡선아래 면적으로 요약한 값.
- (실습 : ROC 와 AUC)



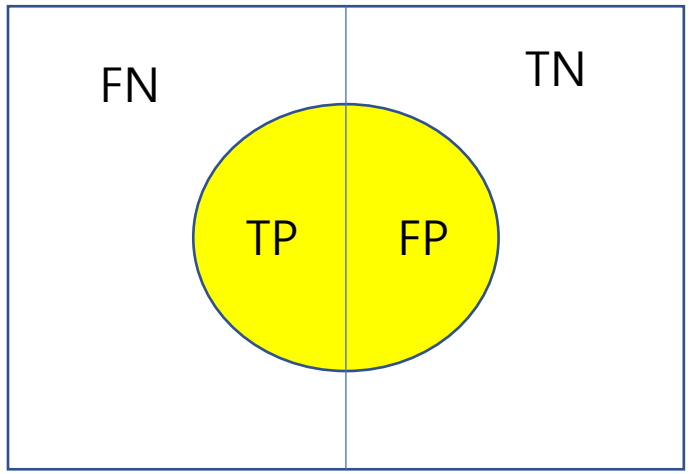


https://en.wikipedia.org/wiki/Receiver_operating_characteristic

ROC 계산 예제

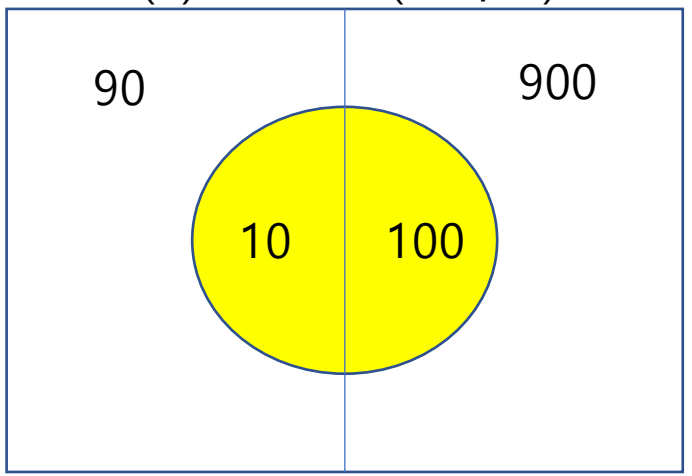
< 데이터 분포 >
'9' 샘플데이터 : 100 개
'9 아님' 샘플데이터 : 1000 개

실제 양성 실제 음성



< Random 예측 >

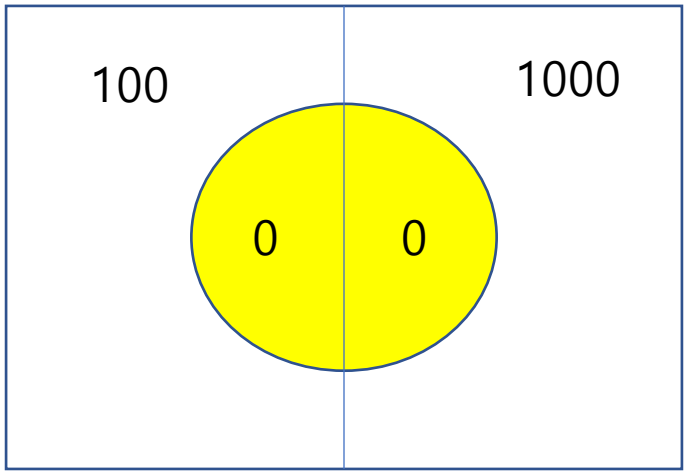
실제 양성 (9) 실제 음성 (9 아님)



FPR = 100/(900 + 100)
TPR = 10/(90 + 10)

< Dummy-최빈값 예측 >

실제 양성 (9) 실제 음성 (9 아님)



FPR = 0/(1000 + 0)
TPR = 0/(100 + 0)

5. ROC곡선을 이용한 최적 성능 조건 파악

- 예제 : 0 ~ 9 숫자 분류 문제에 gamma를 다르게 적용한 SVM 모델
- gamma = [1, 0.1, 0.01]

- (실습)
  - 정확도는 모두 0.9를 나타냄.(decision_function 기본값 0.5 사용한 경우임)
  - gamma = 1.0인 경우 : 무작위 수준
  - gamma = 0.01인 경우 : 완벽한 모델(AUC = 1)

- 왼쪽 위 꼭지점의 임계값을 찾으면 됨.
  - fpr, tpr, threshold 배열에서 최고의 성능(tpr)을 보이는 threshold 값 확인하면 됨.
  - threshold = -0.858 일 때 TPR = 1 달성함.

https://en.wikipedia.org/wiki/Receiver_operating_characteristic

5. 모델 성능평가 – 회귀 평가지표

- R2(결정계수), MSE(Mean Square Error), MAE(Mean Absolute Error)등의 평가지표가 있음.
- 일반적으로 R2가 무난하게 사용됨.
- 변수(실제값-예측값)간 영향을 주는 정도를 정량화한 수치
  - 1에 가까울 수록 실제값-예측값 간 관련성이 큼.
  - 0에 가까울 수록 실제값-예측값 간 관련성이 없음.

5. 모델 성능평가 – 모델선택에서 평가지표 사용하기

- 교차검증, 그리드서치에서 이전에는 정확도만을 사용하였음.
- 기본평가지표는 accuracy(정확도) 이나 불균형 데이타셋에 대한 성능평가가 필요할 때 필요한 평가지표를 지정하여 사용 가능함.
  - cross_val_score() 함수를 이용하여 accuracy, average_precision, roc_auc 등 교차검증시 지정된 지표 측정.
  - cross_validate() 함수를 사용하여 여러가지 성능척도를 한번에 볼 수도 있음.

- (실습) : 9개의 숫자를 분류하는 문제에 SVC 적용
  - 정확도와 정밀도의 차이를 확인하기 위해 일부러 적절하지 않은 매개변수 그리드 적용
  - 지정된 측정척도를 최적화하는 매개변수값을 찾음.
- 측정척도로 정확도 사용했을 때(기본값)
  - 정확도를 사용했을 때 선택된 매개변수는 gamma = 0.001
  - 테스트셋으로 평가해보면 정확도 점수(0.973)가 평균정밀도 점수(0.966)보다 높음.

- 측정척도로 정밀도 사용했을 때
  - 평균 정밀도를 사용하면 gamma = 0.01이 선택됨.
  - 테스트셋으로 평가해보면 평균정밀도 점수(0.996)가 정확도 점수(0.896) 보다 높음.