데이터로 배우는 통계학

자연과학대학 통계학과
장원철 교수

미래를 예측하고 싶다면?
– 알고리즘 알아보기

1. 알고리즘과 인공지능

알고리즘과 인공지능 그리고 데이터 사이언스

- 알고리즘: 데이터를 기반으로 실생활의 문제에 대한 해답을 제공하는 기술
- 머신러닝: 통계학과 컴퓨터 공학의 접점 분야로 경험(데이터)를 통하여 자동적으로 향상되는 컴퓨터 알고리즘을 연구하는 분야
- AI 분야 대표적 석학 마이클 조단 교수는 Harvard Data Science Review에 투고한 “Artificial Intelligence – The Revolution Hasn’t Happened Yet”에서 인공지능(과 데이터 사이언스)는 사실 머신러닝의 또다른 이름이라고 언급

지도학습

- **인공지능(=머신러닝)이 수행하는 작업은 크게 지도학습과 자율학습으로 나누어지며 지도학습은 다시 분류와 예측으로 나누어진다.**
- **지도학습(Supervised Learning)**
  - 분류(Classification): 사진에서 고양이와 개의 분류
  - 예측(Prediction): 주식가격 예측, 날씨 예측

강 인공지능과 약 인공지능

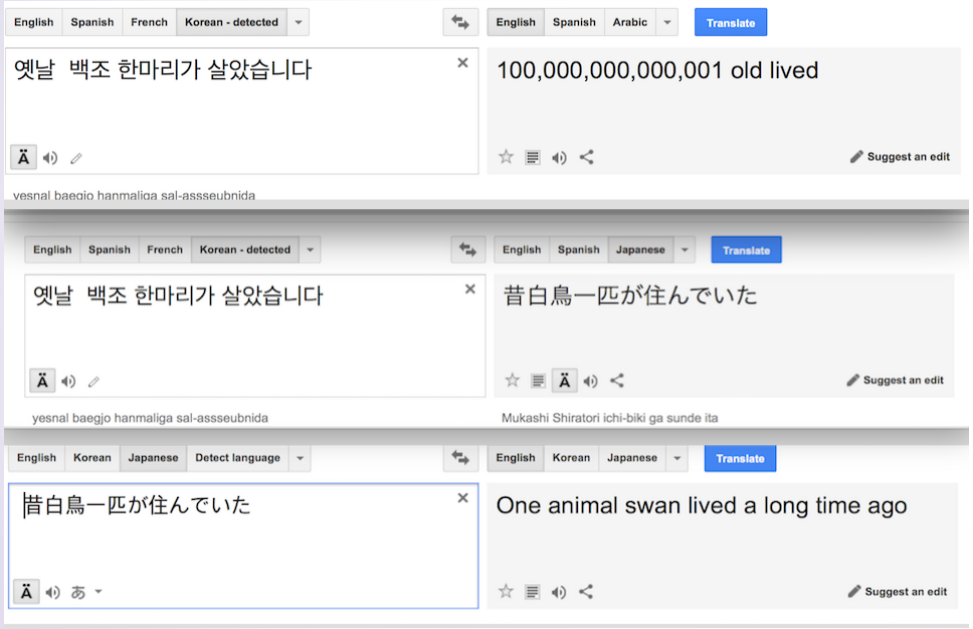
[Terminator 3: Rise of Machines]
(공식포스터)

[Deep Mind]
(Wikipedia)

- 강 인공지능: 자의식을 가진 인공지능으로 명령받지 않은 일도 할 수 있으며 심지어 명령을 거부할 수 있음
- 약 인공지능: 데이터를 바탕으로 주어진 문제에 대한 답을 찾는 기술(예: 알파고)

기타 출처 #1, 2

약 인공지능과 구글 번역기



[구글 번역기]
(Google)

기타 출처 #3

빅데이터의 2가지 유형

- 길쭉한 자료: n이 큰 경우(표본 크기가 큰 경우)
- 뚱뚱한 자료: p가 큰 경우(parameter의 개수가 많은 경우)

길쭉한 자료

월별 신용카드 사용량

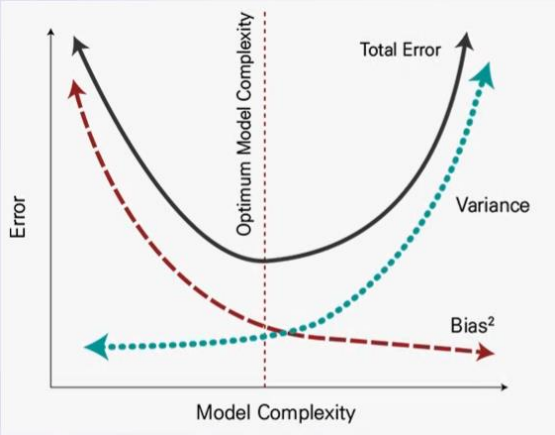
이름	식료품	의류	레저	외식	화장품	
최수지	30,000	500,000	6,000	320,000	730,000	
예린	50,000	400,000	12,000	400,000	490,000	
정국	200,000	80,000	30,000	1,500,000	0	
장원철	500,000	100,000	100,000	1,080,000	50,000	
강수지	60,000	480,000	40,000	210,000	520,000	
배수지	50,000	600,000	400,000	570,000	630,000	
…	…	…	…	…	…	

뚱뚱한 자료

- 마이크로 어레이 - 유전자의 발현정도를 측정하는 바이오칩

이름	유전자 1	유전자 2	유전자 3	…	유전자 20,000
설현 (정상)	0.6025	0.1902	-0.6775	…	-0.3604
혜리 (정상)	-0.1102	-1.0011	0.2501	…	-1.2731
장원철 (간암 환자)	4.6289	0.2102	0.7825	…	0.3691
정하웅 (간암 환자)	3.7501	1.3027	0.3223	…	0.8951

과적합과 bias-variance trade-off



[Bias and variance contributing to total error]
(Understanding the Bias-Variance Tradeoff)

- 만약 주어진 데이터를 모두 사용하여 모형을 적합시킨다면 과적합(overfit)이 생길 수 있다.
- 복잡한 모형을 적합할 경우 예측치의 편의는 줄어들지만 분산은 늘어나고(즉 과적합이 발생) 단순한 모형을 적합할 경우 반대 현상이 일어난다.
- 따라서 적절한 모형을 적합시켜서 과적합을 피해야 한다.

기타 출처 #4

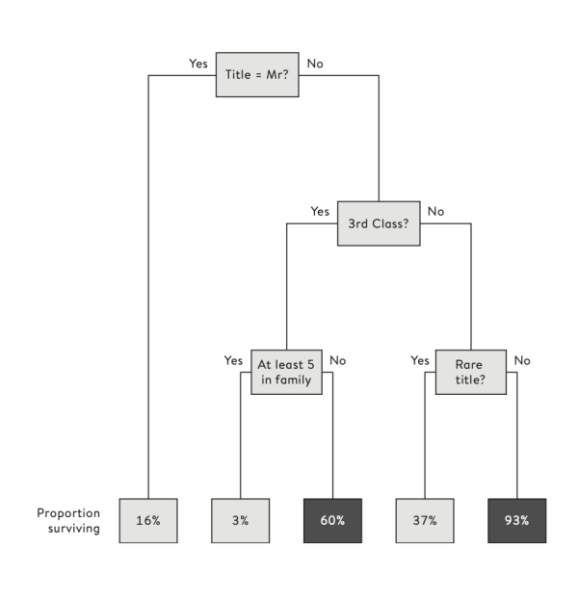
훈련자료, 평가자료, 검증자료

- 과적합을 피하기 위한 방안으로는 전체 데이터를 훈련자료(training data)와 평가자료(test data)로 나눈 후 훈련자료를 이용하여 분석모형을 정하고 제안한 방법의 평가를 위해서 평가자료를 이용할 수 있다.
- 케글과 같은 분석경진대회에서 이러한 방법을 사용하고 있고 cheating을 방지하기 위해 공개용 평가자료와 비공개용 평가자료를 사용한다.
- 모형의 복잡도를 결정하기 위해서 별도의 검증자료(validation data)를 사용하거나 훈련자료를 활용하는 교차검증(cross-validation) 방법이 있다.

사례연구: 타이타닉호의 생존자는 어떤 사람들이었을까?

- 타이타닉호는 첫 항해에서 빙하와 부딪혀서 1912년 4월 14일과 15일 사이에 천천히 침몰하였다.
- 우리는 타이타닉호 승객들의 정보를 이용하여 개별 승객들의 생존확률에 대해서 알고 싶다.
- 총 1,309명의 승객 중 897명의 사례를 훈련자료로, 나머지 412명의 자료를 평가자료로 사용하였다.

의사결정나무



[의사결정나무를 이용한 생존율 예측]
(The Art of Statistics, p155)

- 의사결정나무는 가장 단순한 형태의 알고리즘 중 하나로 일련의 예/아니오 질문을 통해 최종적으로 개별 승객들이 사망할지 여부를 분류할 수 있다.
- 왼쪽 그림에서 끝 마디의 생존율이 50% 이상일 경우 생존자로 분류한다.

기타 출처 #5

오늘의 강의 요점

- 인공지능, 머신러닝, 데이터 사이언스
- 머신러닝의 2가지 분야
  - 지도학습
    - 분류
    - 예측
  - 자율학습

미래를 예측하고 싶다면?
– 알고리즘 알아보기

2. 알고리즘 성능평가

분류 방법의 성능은 어떻게 측정할 수 있을까?

- **주어진 분류 알고리즘의 성능은 어떻게 평가할 수 있을까?**
  편의상 알고리즘이 각 관측치를 양성 또는 음성으로 분류한다고 하자.

  - 정확도(accuracy): 평가자료에서 1-오분류 비율
  - 민감도(sensitivity): 양성환자 중 양성으로 진단된 비율
  - 특이도(specificity): 음성환자 중 음성으로 진단된 비율

Confusion Matrix

	음성 진단	양성 진단
실제 음성	True Negative(TN)	False Positive(FP)
실제 양성	False Negative(FN)	True Positive(TP)

- 정확도: (TP+TN)/(TP+FP+FN+TN)
- 민감도: TP/(TP+FN)
- 특이도: TN/(FP+TN)
- 타이타닉에서 생존자를 예측하는 문제는 질병 검사에서 양성인지를 예측하는 문제와 동일하다. 즉 생존=양성으로 생각할 수 있다.

사례연구 타이타닉호 생존자 예측

훈련자료			
	사망 예측	생존 예측	합계
실제 사망	475	93	568
실제 생존	71	258	329
합계	546	351	897

- 정확도: (475+258)/897=0.82
- 민감도: 258/329=0.78
- 특이도: 475/568=0.84

사례연구 타이타닉호 생존자 예측

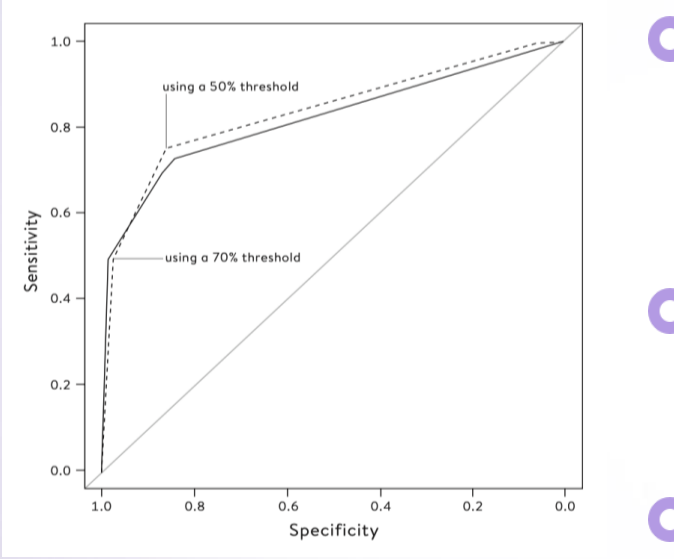
평가자료			
	사망 예측	생존 예측	합계
실제 사망	228	45	273
실제 생존	35	104	139
합계	263	149	412

- 정확도: (228+104)/412=0.81
- 민감도: 104/139 =0.75
- 특이도: 228/273 = 0.84

ROC 곡선

- 의사결정나무에서는 각 승객의 생존확률을 예측한다.
- 생존확률이 50% 이상일 경우 생존자로 분류한 결과 민감도와 특이도가 각각 0.78과 0.84였다.
- 만약 생존자로 분류하는 기준을 보수적으로 잡기 위해 생존확률이 70% 이상일 경우만 생존으로 분류한다면 이 경우 민감도와 특이도는 각각 0.50과 0.98이 된다.
- 이처럼 생존자 분류기준을 변경할 경우 민감도와 특이도의 값이 달라지는데 사람마다 분류기준점이 달라질 수 있으므로 분류기준값에 따른 특이도와 민감도를 제시하는 그림을 ROC 곡선이라고 한다.

ROC 곡선



[훈련자료(점선)과 평가자료(실선)을 이용한 의사결정나무의 ROC 곡선]
(The Art of Statistics, p160)

- 왼쪽그림에서 y축은 민감도를 x축은 특이도를 나타낸다. 다만 x축은 1에서부터 0으로 감소함을 주목하자.
- 일반적으로 기준점을 상향 조정하게 되면 특이도는 증가하고 민감도는 감소한다.
- 여기서 대각선은 동전던지기(앞면이 나오면 생존예측, 뒷면이 나오면 사망예측)와 같이 전혀 쓸모 없는 분류 알고리즘의 ROC 곡선을 나타낸다.

기타 출처 #6

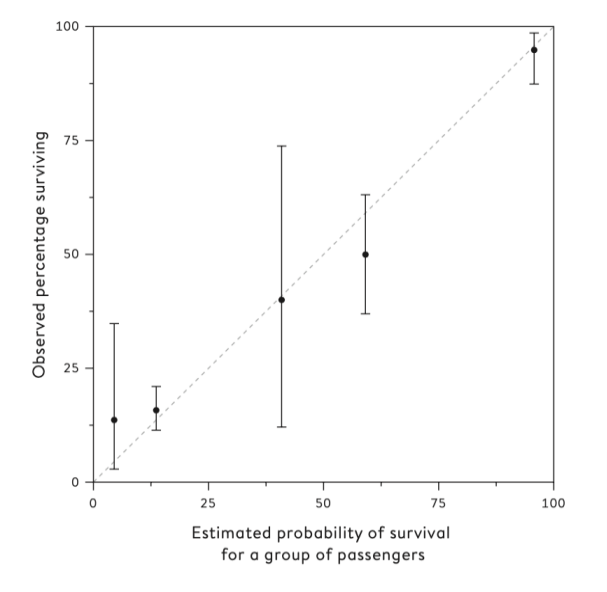
ROC 곡선의 비교와 AUC

- 만약 두개의 다른 분류 알고리즘을 비교하고자 할 때 각 분류 알고리즘의 ROC 곡선을 이용하면 된다.
- 두개의 ROC 곡선이 겹치지 않는 경우 위에 있는 ROC 곡선에 해당하는 분류 알고리즘이 우수하다. 즉 같은 기준점에서는 해당 알고리즘의 특이도와 민감도가 더 높다는 걸 의미하기 때문이다.
- 하지만 많은 경우 ROC곡선들은 겹쳐서 표시되기 때문에 이 경우는 ROC곡선 아래면적(AUC)을 이용하여 분류 알고리즘들을 비교할 수 있다.
- 동전던지기와 같은 분류 알고리즘은 AUC값이 0.5이며 완벽한 분류 알고리즘이 있다면 AUC는 1이다.

기상예보에서 비가 올 확률은 무얼 의미하는 걸까?

- ROC 곡선은 분류 알고리즘 자체에 대한 평가를 하는데 사용할 수 있지만, 각각 생존확률 예측치가 얼마나 정확한지는 알려주지 않는다.
- 이런 확률적 예측에 가장 민감한 사람들은 기상예보관이다!
- 기상예보는 현재 조건하에서 날씨가 어떻게 변할지 수학적으로 계산하는 복잡한 모형을 통해서 가상의 내일 날씨를 여러 번 생성해낸다. 예를 들어 50개의 가상 내일 날씨를 생성했는데 이 중 5번이 비가 왔다고 하면 내일 비가 올 확률은 10%가 되는 것이다.

보정그림(Calibration Plot)

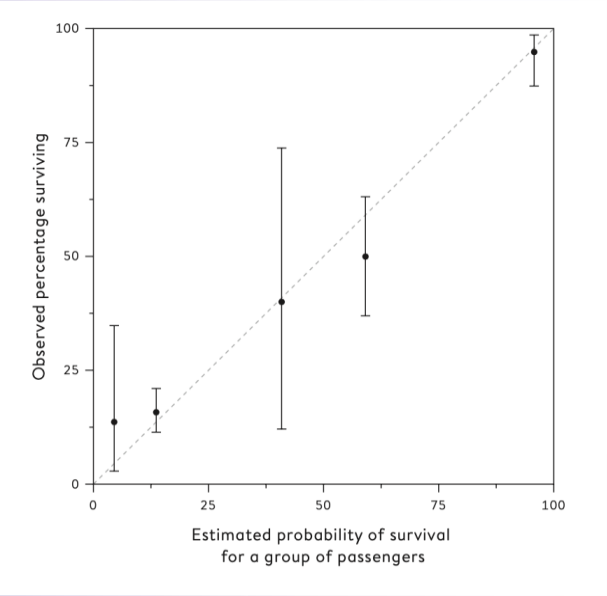


[의사결정나무 보정그림]
(The Art of Statistics, p162)

- 확률예측이 얼마나 정확했는지 알아보기 위해 보정그림을 고려할 수 있다.
- 앞에서 의사결정나무를 이용하여 전체 승객을 5개의 그룹으로 나누었으며 그룹 안에 속한 사람들의 생존확률은 동일하였다.

기타 출처 #7

보정그림(Calibration Plot)



[의사결정나무 보정그림]
(The Art of Statistics, p162)

- 보정그래프는 각 그룹의 생존확률과 더불어 불확실성을 표시하는 구간을 같이 제시한다.
- 여기서 각 구간들이 대각선 점을 포함하기를 기대한다.

기타 출처 #7

브라이어 지수(Brier Score)

- ROC 곡선은 알고리즘 전체의 성능평가를 위해, 보정그림은 개별확률 예측의 불확실성을 알아보기 위해 사용한다.
- 이 두 가지 정보를 모두 통합해서 하나의 측도로 제시할 수 있을까?
- 다행히 이 질문에 대한 대답은 1950년 기상학자 글렌 브라이어가 제공하였다.
- 예를 들어 비가 올 확률이 0.7이라고 할 때 실제 비가 온 경우를 1, 그렇지 않은 경우를 0으로 간주하여 비가 올 확률과 실제 비가 온 지 여부의 차이(오차)의 제곱 구한 후 이렇게 구한 제곱 오차들의 평균을 브라이어 지수라고 한다.

브라이어 지수(Brier Score)

- 브라이어 지수를 이용하여 기상예보를 잘하는지 평가하기 위해서 기준지수(reference score)과 비교할 수 있다.
- ROC 곡선에서 동전던기기가 비교기준이었던 것처럼 여기서 기준지수는 예측하고자 하는 날짜의 과거 기상기록을 이용해서 강수확률을 예측한 후 그 결과에 대한 브라이어 지수를 의미한다.
- 즉 과거 기록을 보고 이번 주 주중(월-금)에 비가 올 확률이 20%라고 가정하고 실제 비가 온 지 여부와 비교하여 기준 기수를 계산할 수 있다. 만약 수요일과 목요일에만 비가 왔다면 이 경우 기준지수는 0.28이다.

기술지수(Skill Score)

- 제대로 된 기상예보 알고리즘이라고 하면 그 알고리즘의 브라이어 지수는 기준지수보다 낮아야 한다.
- 기준지수 대비 해당 알고리즘이 얼마나 오차를 감소시켰는지 알아내는 지표를 기술지수라고 하면 1-(해당 알리고즘의 브리이어 지수)/기준지수로 정의할 수 있다.
- 오늘날 강수 예보 시스템의 기술지수는 다음날에 대해서는 대략 0.4이고 향후 일주일에 대해서는 0.2이다.

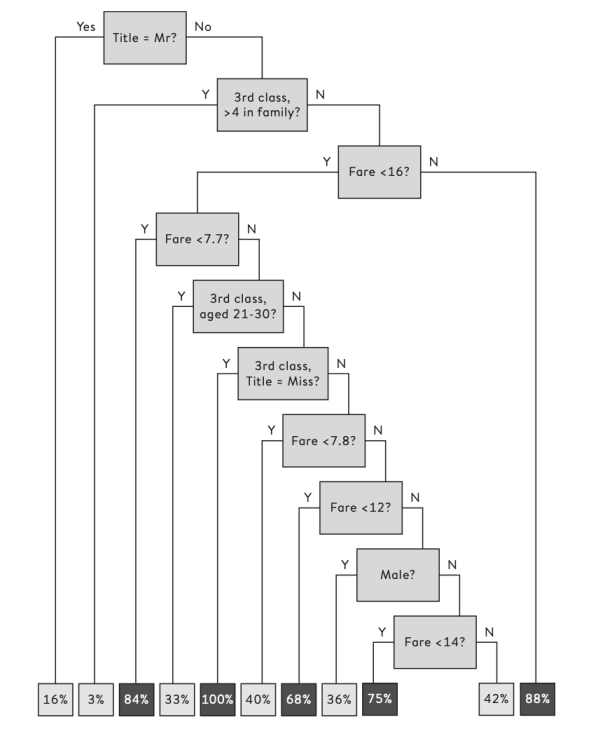
오늘의 강의 요점

- **분류 방법의 성능평가**
  - Confusion matrix
  - ROC 곡선
  - 브라이어 지수

미래를 예측하고 싶다면?
– 알고리즘 알아보기

3. 과적합과 알고리즘의 문제점

과적합

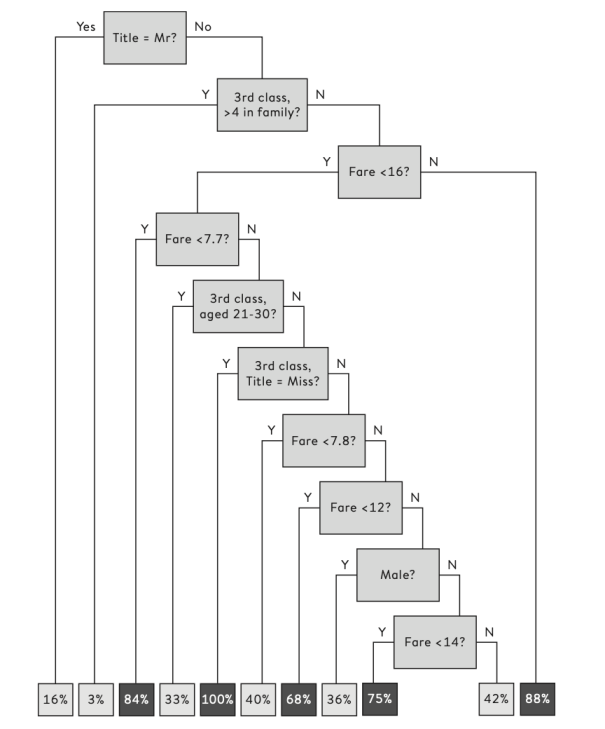


[과대적합된 의사결정나무의 예]
(The Art of Statistics, p168)

- 의사결정나무에서 계속 조건을 추가할 경우 마지막 끝마디의 개수가 늘어난다. 즉 모형의 복잡도가 증가하면서 과적합이 일어난다.
- 왼쪽 그림의 의사결정나무의 훈련자료에서는 정확도는 0.83으로 이전 의사결정나무의 훈련자료에서의 정확도보다 높다.

기타 출처 #8

과적합



[과대적합된 의사결정나무의 예]
(The Art of Statistics, p168)

- 하지만 평가자료에 적용 시 정확도는 0.81로 떨어지며 브라이어 지수의 값은 0.150으로 이전 의사결정나무의 브라이어 지수 0.139보다 높다.

기타 출처 #8

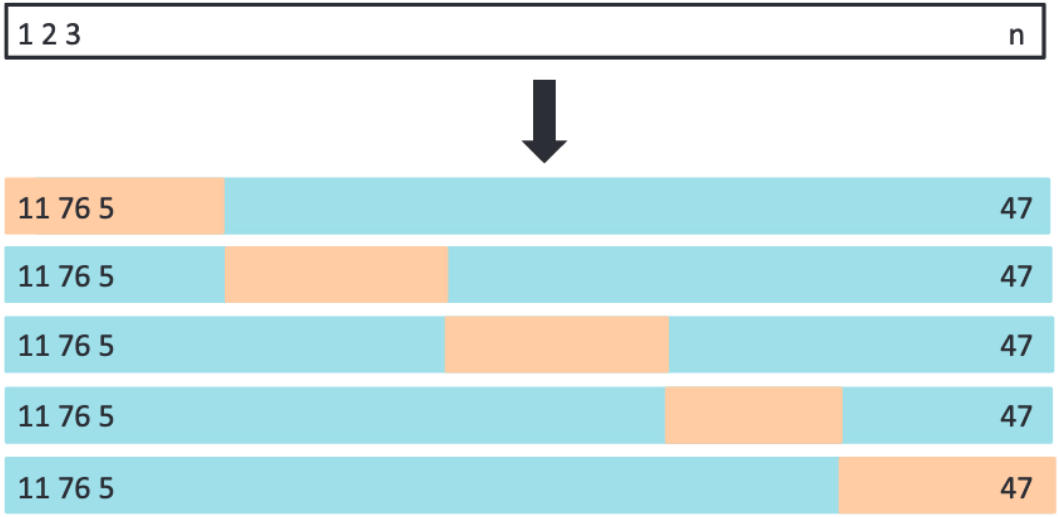
교차검증

- 과적합을 피하기 위해 적절한 복잡도를 가진 모형을 선택해야 한다.
- 이를 위해서 훈련자료와 별도로 모형의 복잡도를 결정하기 위한 검증자료가 별도로 필요하다.
- 보통 모형의 복잡도는 조절모수(tuning parameter)를 통해 결정되는데 각각 다른 조절모수의 값을 이용해서 검증자료에서 예측오차를 계산한 후 그중 가장 작은 예측오차를 제공하는 조절모수를 선택하는 방식이다.
- 의사결정나무의 경우 조절모수는 가지(조건)의 개수로 생각할 수 있다.

K-fold 교차검증

- 1개의 교차자료만으로 조절모수를 결정할 경우 어떤 자료를 교차자료로 사용했는지에 따라 조절모수의 선택에 민감한 영향을 줄 수 있다.
- 이런 단점을 보완하기 위해 많이 사용하는 방법이 K-fold 교차검증이다.
- 전체 데이터를 K개로 나눈 후 그중 하나를 검증자료로 나머지 K-1개의 자료를 훈련자료로 사용한다. 이런 과정을 검증자료를 바꿔가며 K번 반복한다.
- 결론적으로 K개의 예측오차를 계산할 수 있고 최종적으로 이 값들의 평균을 교차자료에서는 예측오차로 사용한다.
- 일반적으로 K=5 또는 10을 사용한다.

K-fold 교차검증



[A schematic display of 5-fold CV]
(Introduction to Statistical Learning, p181)

기타 출처 #9

그 외 다양한 분류방법들

- Random Forest
- Support Vector Machine(SVM)
- Neural Network
- K-Nearest Neighbor(KNN)

알고리즘의 문제

- Robustness 이슈
- 변동성에 대한 고려
- 내재적 편향성
- 투명성
- Reverse-engineering

Robustness 이슈 – 구글독감예측 사례

- 2008년 구글이 독감예측을 위해 Google Flue Trend를 개발
- 사람들이 독감증상이 있을 경우 구글을 이용해 독감관련 검색을 한다는 사실에 착안해서 특정단어들의 검색량을 이용하여 독감환자 예측
- 처음에는 잘 맞았지만 2013년 발병율을 2배 넘게 예측하면서 신뢰도 하락으로 서비스 중단
- 다른 알고리즘처럼 모든 환경이 똑같이 유지된다는 가정이 있지만 구글자체 검진엔진의 지속적 변화를 반영하지 못했음, 즉 조금의 변화에도 민감하게 예측치가 변화할 수 있음

변동성에 대한 고려

- 한국에서 각종 암발생율 지도를 살펴보면 최상위군과 최하위군은 주로 작은 군으로 구성되어 있다.
- 이유는 사실 인구가 작은 지역자치단체에서는 약간의 변동만으로 크게 순위가 변할 수 있기 때문이다.
- 따라서 작은 숫자에 기반한 예측치는 변동의 가능성이 크다는 점을 고려해야 한다.

내재적 편향성

[시베리아 허스키견]
(Wikipedia)

- 시베리아 허스키견과 늑대사진을 분류하도록 훈련된 시각 인식 알고리즘이 유독 반려견으로 길러진 허스키견을 분별하지 못함
- 알고리즘은 사실 배경의 눈을 기준으로 두 개를 구별하고 있었던 사실이 밝혀짐
- 알고리즘은 연관성을 기반하기 때문에 실제 관심사항에 무관한 특징을 사용할 수 있음

기타 출처 #10

투명성 여부

- 미국 법정에서는 범죄재발예측 알고리즘을 이용하여 계산된 위험지수를 보호관찰 혹은 형량 결정에 참고하는 경우가 있다.
- 이 알고리즘이 어떤 방식으로 위험지수를 산출하는지는 알려져 있지 않다.
- 다만 양육환경과 과거 범죄 연루에 관한 정보를 사용한다고 알려져 있는데 이런 정보를 이용할 경우 사회적 빈곤층의 위험지수가 높아질 가능성이 상당히 크다.

Reverse-engineering

- 이러한 투명성의 부족을 해결하기 위한 방안으로 reverse-engineering을 어느 정도 선에서 허용한다.
- 예를 들면 자동차보험은 성별 차별을 해서는 안 된다. 이 경우 모든 정도를 동일하게 집어넣고 성별만 바꿀 경우 예측치가 변화하는지 알아보는 것을 고려할 수 있다.
- 위와 같은 과정을 reverse-engineering이라고 부른다.
- 이러한 문제점을 해결하기 위해 설명 가능한 인공지능 (Explainable A.I.)가 등장했다.

오늘의 강의 요점

- 과적합
- 알고리즘의 문제점

미래를 예측하고 싶다면?
– 알고리즘 알아보기

Lab 7 사례연구: 타이타닉 승객들의 생존확률

타이타닉 승객 중 누가 생존확률이 높았을까?

- 케글은 데이터 분석 경진대회를 대행해주는 플랫폼이다. 기업이나 공공기관이 데이터로 해결해야 할 문제를 케글에 등록하면 원하는 사람은 누구나 참가해서 과제에 자료분석 결과를 제출할 수 있다.
- 케글에서 2012년부터 2015년까지 타이타닉 승객들의 생존확률을 예측하는 경진대회를 개최하였다.
- 경진대회 홈페이지(https://www.kaggle.com/c/titanic/data)에서 897명으로 이루어진 훈련 자료와 412명으로 이루어진 평가자료를 다운로드 받을 수 있다.

분석 절차

1. 데이터 전처리

2. 의사결정나무 적합

3. Confusion Matrix와 ROC 곡선을 이용한 결과 평가

* 분석에 사용된 code는 https://bit.ly/32M4y4j 을 참조하였다.

데이터 전처리: 데이터 읽기

```
# read data
train <-read_csv('data/titanic/train.csv')
test <-read_csv('data/titanic/test.csv')

# summary statistics for whole data

titanic<-bind_rows(train,test)
summary(titanic)
```

데이터 요약

```
> summary(titanic)
  PassengerId      Survived          Pclass          Name               Sex                 Age        
 Min.   :   1   Min.   :0.0000   Min.   :1.000   Length:1309        Length:1309        Min.   : 0.17  
 1st Qu.: 328   1st Qu.:0.0000   1st Qu.:2.000   Class :character   Class :character   1st Qu.:21.00  
 Median : 655   Median :0.0000   Median :3.000   Mode  :character   Mode  :character   Median :28.00  
 Mean   : 655   Mean   :0.3838   Mean   :2.295                                         Mean   :29.88  
 3rd Qu.: 982   3rd Qu.:1.0000   3rd Qu.:3.000                                         3rd Qu.:39.00  
 Max.   :1309   Max.   :1.0000   Max.   :3.000                                         Max.   :80.00  
                NA's   :418                                                            NA's   :263    
     SibSp            Parch           Ticket               Fare            Cabin              Embarked        
 Min.   :0.0000   Min.   :0.000   Length:1309        Min.   :  0.000   Length:1309        Length:1309       
 1st Qu.:0.0000   1st Qu.:0.000   Class :character   1st Qu.:  7.896   Class :character   Class :character  
 Median :0.0000   Median :0.000   Mode  :character   Median : 14.454   Mode  :character   Mode  :character  
 Mean   :0.4989   Mean   :0.385                      Mean   : 33.295                                        
 3rd Qu.:1.0000   3rd Qu.:0.000                      3rd Qu.: 31.275                                        
 Max.   :8.0000   Max.   :9.000                      Max.   :512.329                                        
                                                     NA's   :1                                              
```

데이터 전처리: 새로운 변수 생성(호칭)

```r
# Grab passenger title from passenger name
titanic$Title <- gsub("^.*, (.*?)\\..*$", "\\1", titanic$Name)

# Frequency of each title by sex
table(titanic$Sex, titanic$Title)
```

```
##
##          Capt Col Don Dona  Dr Jonkheer Lady Major Master Miss Mlle Mme
##   female    0   0   0    1   1        0    1     0      0  260    2   1
##   male      1   4   1    0   7        1    0     2     61    0    0   0
##
##           Mr Mrs  Ms Rev Sir the Countess
##   female   0 197   2   0   0            1
##   male   757   0   0   8   1            0
```

데이터 전처리: 새로운 변수 생성(호칭)

```r
# First, I reassign few categories
titanic$Title[titanic$Title == 'Mlle' | titanic$Title == 'Ms'] <- 'Miss'
titanic$Title[titanic$Title == 'Mme']  <- 'Mrs'

# Then, I create a new category with low frequency of titles
Other <- c('Dona', 'Dr', 'Lady', 'the Countess','Capt', 'Col', 'Don', 'Jonkheer',
'Major', 'Rev', 'Sir')
titanic$Title[titanic$Title %in% Other]  <- 'Other'

# Let's see if it worked
table(titanic$Sex, titanic$Title)
```

```
##
##          Master Miss  Mr Mrs Other
##   female      0  264   0 198     4
##   male       61    0 757   0    25
```

데이터 전처리: 새로운 변수 생성(가족 숫자)

```
FamilySize <- titanic$SibSp + titanic$Parch + 1

table(FamilySize)
```

```
## FamilySize
##   1   2   3   4   5   6   7   8  11
## 790 235 159  43  22  25  16   8  11
```

ere are nine family sizes: 1 to 8 and 11. As this is too many categories, let's collapse some categories as llows.

```
# Create a family size feature with three categories
titanic$FamilySize <- sapply(1:nrow(titanic), function(x)
                        ifelse(FamilySize[x]==1, "Single",
                        ifelse(FamilySize[x]>4, "Large", "Small")))

table(titanic$FamilySize)
```

```
##
##  Large Single  Small
##     82    790    437
```

데이터 전처리

- 새롭게 생성한 Family Size와 Title을 포함하는 훈련 자료와 평가자료를 만든다.

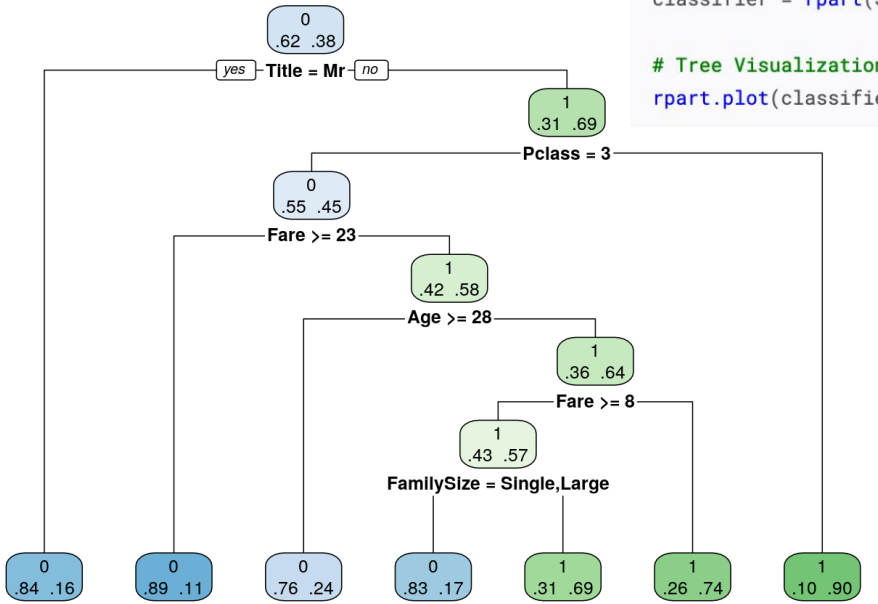
```
# Splitting the dataset into the Training set and Test set
train_original <- titanic[1:891, c("Survived","Pclass","Sex","Age","SibSp","Parc
h","Fare","Embarked","Title","FamilySize")]
test_original <- titanic[892:1309, c("Pclass","Sex","Age","SibSp","Parch","Far
e","Embarked","Title","FamilySize")]
```

데이터 전처리

- 평가자료에는 생존여부가 포함되어 있지 않기때문에 분류 알고리즘의 성능을 평가하기 위한 별도의 검증 자료를 생성한다.

```
# Splitting the Training set into the Training set and Validation set
set.seed(789)
split = sample.split(train_original$Survived, SplitRatio = 0.8)
train = subset(train_original, split == TRUE)
test = subset(train_original, split == FALSE)
```

의사결정나무의 적합




```r
# Fitting Decision Tree Classification Model to the Training set
classifier = rpart(Survived ~ ., data = train, method = 'class')

# Tree Visualization
rpart.plot(classifier, extra=4)
```

의사결정나무 성능평가: Confusion Matrix

```
# Predicting the Validation set results
y_pred = predict(classifier, newdata = test[,-which(names(test)=="Survived")], ty
pe='class')

# Checking the prediction accuracy
table(test$Survived, y_pred) # Confusion matrix
```

```
##    y_pred
##       0   1
##   0 102   8
##   1  21  47
```

```
error <- mean(test$Survived != y_pred) # Misclassification error
paste('Accuracy',round(1-error,4))
```

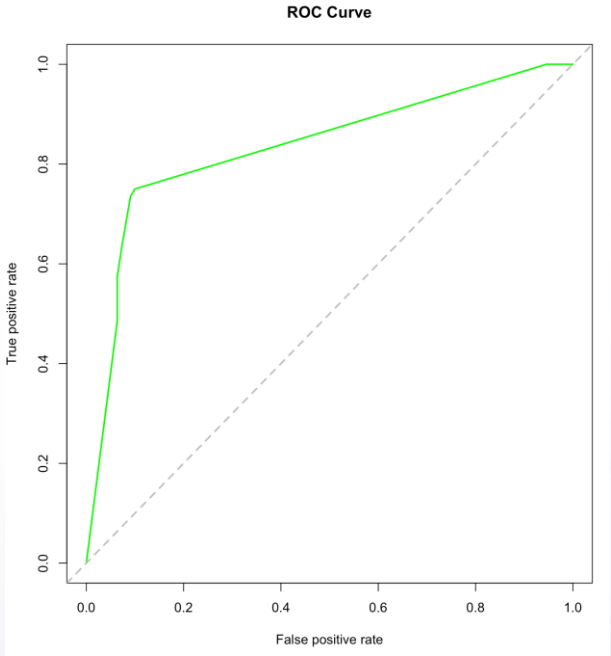
```
## [1] "Accuracy 0.8371"
```

의사결정나무 성능평가: ROC 곡선

```
# ROC curve
prob_pred = predict(classifier, newdata = test[,-which(names(test)=="Survived")], type='prob')
fitpred = prediction(prob_pred[,2], test$Survived)
fitperf = performance(fitpred,"tpr","fpr")
plot(fitperf,col="green",lwd=2,main="ROC Curve")
abline(a=0,b=1,lwd=2,lty=2,col="gray")

# AUC calculation
dt_auc <- performance(fitpred, measure = "auc")
titanic_dt_auc <-dt_auc@y.values[[1]]
titanic_dt_auc
```

```
[1] 0.8344251
```



오늘의 강의 요점

- 타이타닉 승객의 생존율 예측 분석
- 분석에 사용된 R code는 https://github.com/wcjang/K-MOOC 에서 다운로드 받을 수 있다.

출처

#1 namuwiki https://namu.wiki/jump/YX%2BBfifz%2Fxm8RA4%2BDkR5trEpCibKTm3Z1uztOYsxefQ4vtLbETGqjFd2Xw4S8FqhQK%2FlvtNYFxF%2BzisaqwgWipFP6p9DjlAZAHJ9eMSBuPvI7LwwN%2BvFHXpXiWMdN%2BPM

#2 https://bit.ly/3lrVAAA

#3 https://translate.google.co.kr/?hl=ko

#4 https://bit.ly/2IvdCDl

#5~8 D. Spiegelhater, (2019), The Art of Statistics, Penguin Random House

#9 James, Witten, Hastie and Tibshirani (2013), Introduction to Statistical Learning, Springer

#10 Wikipedia https://bit.ly/3lxAAZ5