[핵심01] 인공지능(intelligence)의 정의

인간이 가진 지능과 **유사한 기능**을 갖도록 하는 **컴퓨터 프로그램**입니다.

(1) learning ability: 학습 능력
(2) reasoning ability: 추론 능력
(3) perception ability: 인지 능력
(4) act: 행동

[핵심02] 기계학습의 정의 by Mitchell

어떤 **컴퓨터 프로그램**이 **T**(분류)라는 **작업**을 **수행**한다. 이 **프로그램**의 **성능**을 **P**(인식률)라는 척도로 **평가**했을 때, **경험 *E*(학습)를 통해 성능이 개선된다면** 이 프로그램은 **학습을 한다**고 말할 수 있습니다.

T = 작업(분류) #회귀분석,...
E = 경험(학습) #데이터
P = 성능평가척도(인식률)

[핵심03] **퍼셉트론의 한계**

퍼셉트론은 **결정초평면 역할을 하는 선형분류기**입니다. **선형 분리가 불가능한 상황에 대처하지 못합니다.**

[핵심04] **일반화 능력**

가능한 **작은 바이어스**와 **낮은 분산**을 가지도록 **모델을 선택하는 것**이 필요합니다.
일반화 능력이 좋은 모델 - 학습데이터와 테스트 데이터 모두에게 성능이 좋은 모델

[핵심05] **검증데이터** # 모델을 비교할 때 사용하는 데이터입니다.

좋은 모델을 알고 있는 경우라면 바로 테스트 데이터를 적용하면 됩니다.
좋은 모델을 모르는 경우라면 (대부분의 경우 입니다.)
[0] for 모델 in 모델집합 (--> **여러 모델들을 학습**시킨 후)
[1] **검증데이터**로 학습된 모델의 **성능**을 **측정**하고
[2] **가장 높은 성능을 보인 모델**을 선택합니다.
[3] **테스트데이터**로 모델의 **일반화능력(성능)**을 **측정**합니다.

[핵심06] **K-교차검증 #k=개수**

대부분 경우 데이터를 구하는 일 조차 힘듭니다. <-- 데이터의 수집은 비용이 많이 듦 이를 해결하기 위해 K-교차검증을 사용합니다.
[0] **학습데이터를 k개 그룹으로 분할합니다.**
for 모델 in 모델집합
  for i in range(k)
[1] 모델을 **학습데이터(i그룹제외)로 학습**시키고
[2] 학습된 모델의 **성능**을 **i그룹 데이터**로 **측정**합니다.
[3] k개 성능을 평균하여 모델의 성능으로 정합니다.
[4] **가장 높은 성능을 보인 모델**을 선택합니다.
[5] **테스트데이터**로 모델의 **일반화능력(성능)**을 **측정**합니다.

[핵심07] 규제의 종류

규제는 충분히 크고 깊은 모델을 선택한 후, 각종 규제를 적용하여, 기계학습이 높은 일반화 능력을 확보하게 합니다.
명시적 규제로는 **가중치 감쇠**, **드롭아웃**이 있고 **암시적 규제**로는 **조기멈춤**, **데이터 확대**, **앙상블**이 있습니다.

[핵심08] 행렬의 곱셈

행렬 A의 열과 B의 행의 개수가 같아야 합니다.

$$c_{ij} = \sum_{k=1,s} a_{ik} b_{kj}$$

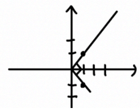
2*3 행렬 $\mathbf{A} = \begin{pmatrix} 3 & 4 & 1 \\ 0 & 5 & 2 \end{pmatrix}$와 3*3행렬 $\mathbf{B} = \begin{pmatrix} 2 & 0 & 1 \\ 1 & 0 & 5 \\ 4 & 5 & 1 \end{pmatrix}$을 곱하면 2*3 행렬 $\mathbf{C} = \mathbf{AB} = \begin{pmatrix} 14 & 5 & 24 \\ 13 & 10 & 27 \end{pmatrix}$

[핵심08] **고유벡터와 고유값**
어떤 행렬 A에 대하여 벡터(v)를 곱했더니, 람다와 같아졌습니다. v는 고유벡터입니다.
고유벡터(v)는 <u>벡터의 방향</u>을 나타내고,
고유값(람다)은 <u>벡터의 길이</u>를 나타냅니다.

(예) $\mathbf{A} = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$

벡터의 길이 : λ,람다,고유값

- $\begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}\begin{pmatrix} 1 \\ 1 \end{pmatrix} = 3\begin{pmatrix} 1 \\ 1 \end{pmatrix}$ $\lambda = 3, \mathbf{v} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$
- $\begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}\begin{pmatrix} 1 \\ -1 \end{pmatrix} = 1\begin{pmatrix} 1 \\ -1 \end{pmatrix}$ $\lambda = 1, \mathbf{v} = \begin{pmatrix} 1 \\ -1 \end{pmatrix}$



[핵심10] 베이즈 정리
어떤 두 일이(x, y)이 동시에 일어날 확률을 결합확률이라고 합니다.
x와 y가 동시에 일어날 결합확률과 **y와 x가 동시에 일어날 결합확률**이 <u>**같습니다**</u>.
<u>**(1) 사전확률과 (2) 우도를 구할 수 있다면, 사후확률을 간접적으로 계산할 수 있습니다.**</u>

$$P(x|y) = \frac{P(y|x)P(x)}{P(y)}$$

[핵심09] **교차 엔트로피(cross-entropy)**
<u>두 확률분포간의 차이를 수치화시키는 것입니다.</u>
두 확률분포간의 차이 = -((P에 대한 확률분포 * Q의 자기정보량)의 총합) --> 차이가 없으면 0입니다.

$$E(P,Q) = -\sum_{x} P(x)\log_2 Q(x) = -\sum_{i=1,k} P(e_i)\log_2 Q(e_i)$$

[핵심10] 기계학습의 최적화 이론
학습데이터에 따라 정해지는 **목적함수**의 **최저점**을 탐색 (# 목적함수 = 미분가능한 함수)하려면 **모든 지점**에서 **순간변화율**을 알아야 합니다.
(1) 미분하는 과정이 필요(오차역전파)
(2) 주로 스토캐스틱 경사하강법을 사용

[핵심10] 경사하강 알고리즘
배치 경사하강 알고리즘 - [0] 샘플의 gradient를 평균하고, [1] 한꺼번에 갱신
<u>스토캐스틱 경사하강 알고리즘 - [0] 한 샘플의 gradient를 계산하고, [1] 즉시 갱신</u>

[핵심11] 델타규칙
단층 신경망의 학습 방법 중 하나입니다.
경사하강법을 이용하여 손실함수의 최소값을 찾아냅니다. --> 이것은 최적의 가중치를 찾는 방법입니다.
=> 입력된 데이터들로 매개변수(가중치, 편향)을 자동으로 결정합니다.

[핵심12] 손실함수
학습이 얼마나 잘 되었는지 확인하기 위해 에러를 측정하여 지표에 나타냅니다.
(1) MSE
(원래 데이터 - 측정 데이터)를 제곱하여 더한 총 값을 N으로 나눠줍니다.
(2) Croess entropy
측정데이터의 확률분포 * 원래데이터의 자기정보량을 곱해준 총 값을 음의 처리 합니다.
오차가 클 수록 x -> 0: y -> 무한대로 가고
오차가 작을 수록 x -> 1: y -> 0에 가깝게 표기되기 때문에
MSE보다 두 확률분포의 차이를 더 잘 확인할 수 있습니다. --> 더 많이 사용됩니다.

[핵심13] 경사하강법
[0] 임의의 곳에서 시작합니다.
[1] (학습률 * 손실함수의 미분값)을 음의처리한 방향으로 갑니다.
*음의 처리한 방향으로 가는 이유는?
f(x)는 양의 방향으로 기울기 값이 커지는 쪽으로 알려주기 때문에 최소값을 찾기 어렵습니다.
따라서 그 방향의 반대방향인 -f(x)로 가주면 최저점을 찾을 수 있습니다.
[2] cost값이 최소가 되는 w를 구합니다.

[핵심14] 수치미분의 한계 극복 - 오차역전법
기울기를 효율적으로 계산할 수 있습니다.
<1> 합성함수: 합성함수를 구성하는 각 함수의 미분의 곱입니다.
<2> 연쇄법칙: 국소미분으로, 각 노드는 자신과 관련된 계산만 수행하고 그외의 노드 계산에는 신경쓰지 않습니다.
<3> 함수의 미분: 합성함수를 이용하여 잘 알아차려야 합니다.
<4> 계산
덧셈: 이전의 미분값을 그래도 흘러보내줍니다. --> 순방향 입력신호 값 불필요
곱셈: 이전의 미분값에 * 입력신호를 바꾼 값을 곱하여 흘러보내줍니다. --> 순방향 입력신호 값 필요

=> 계산그래프
순방향 미분: 모든 노드에 대해 e의 미분을 별도로 계산해야 합니다. --> 노드가 많아지면 힘듭니다.
역방향 미분: 모든 노드에 대한 e의 미분값을 극소미분으로 전달합니다. --> 효율적