eXplainable Artificial Intelligence (XAI):

설명 가능한 인공지능

https://youtu.be/UnzrIAa07DE

XAI의 정의

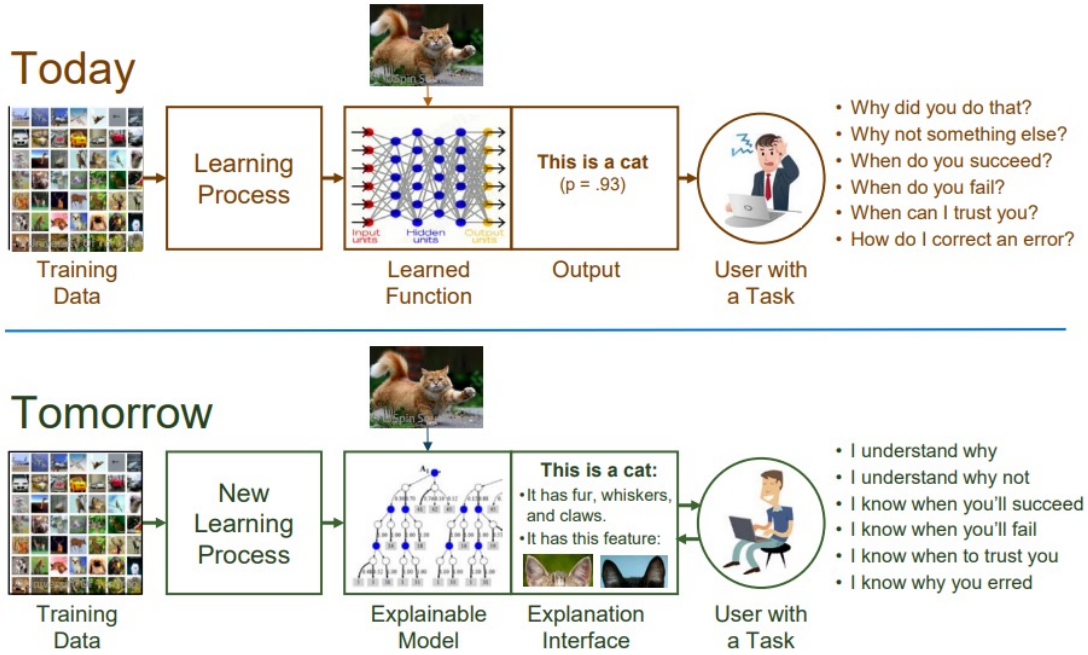
XAI의 필요성

XAI 알고리즘

실습

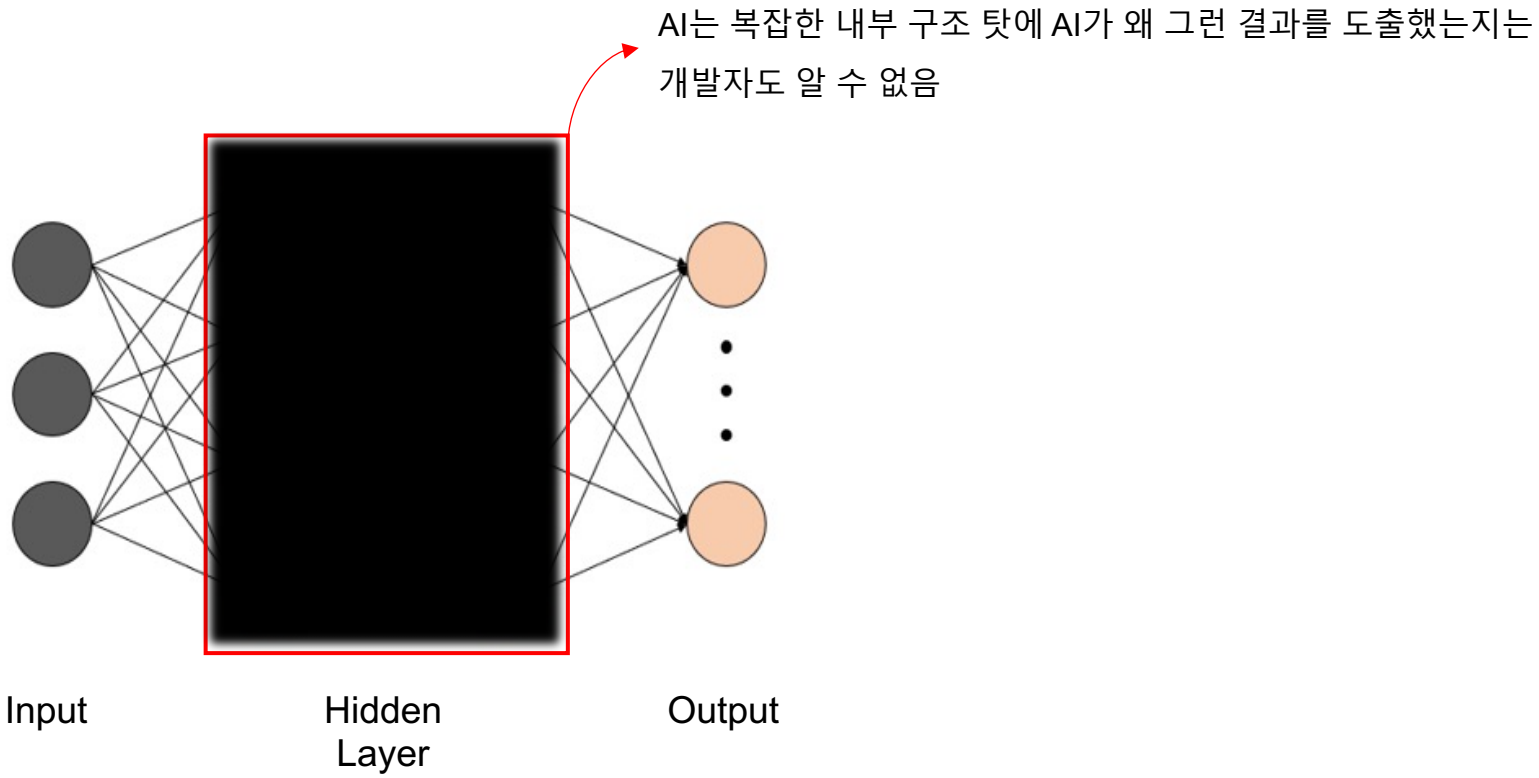
XAI란?

- XAI
  - 사람이 AI가 도출한 결과에 대해 이해할 수 있고 해석이 가능한 AI
  - 즉, 결과에 대해 설명 가능하도록 해주는 기술



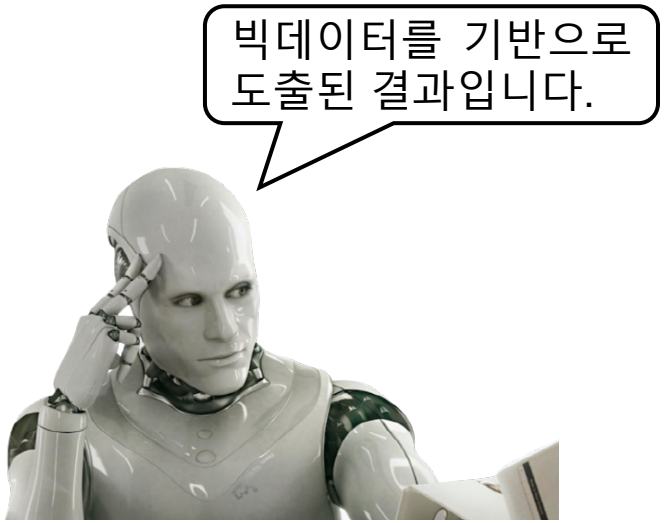
XAI의 필요성

- XAI의 필요성
  - 딥러닝은 추출부터 판단까지 전부 네트워크가 알아서 하므로 결과에 이르기까지의 과정을 확인할 수 없음
  - 이러한 이유로 딥러닝을 Black box 모델이라고도 함



XAI의 필요성

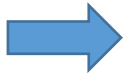
- XAI의 필요성
  - AI를 통해 대출 가능 여부를 판단하는 은행에서 대출이 거절될 경우, 어떠한 문제 때문에 거절됐는지에 대한 설명을 요구한다면 은행측은 빅데이터를 기반으로 판단했다고밖에 말할 수 없음
  - AI가 특정 파일을 랜섬웨어라고 판단할 경우, 왜 그런 결과를 도출했는지는 알 수 없음 (신뢰도 하락)
- 지금까지의 딥러닝은 대부분 결과에 대한 근거를 댈 수 없었음
- 결과가 어떤 과정을 통해 도출됐는지에 대한 설명이 있다면
  - 결과에 대해 더 잘 받아들이고 신뢰도가 높아짐
  - 사용자가 AI의 결정을 이해할 수 있음



Shapley Value

- Shapley Value란?
  - 특정 변수가 결과에 얼마나 영향을 끼치는지 파악하기 위한 수치
  - 예제) 민성, 시현, 대현, 예준 4명이 힘을 합쳐 하루 동안 풀 수 있는 알고리즘 문제 개수는?
    - 한 명도 빠짐 없이 다 같이 풀었을 경우 : 50문제
    - 민성이만 빠지고, 시현, 대현, 예준 셋이 풀었을 경우 : 10문제
    - 시현이만 빠지고, 민성, 대현, 예준 셋이 풀었을 경우 : 15문제
    - 대현이만 빠지고, 시현, 민성, 예준 셋이 풀었을 경우 : 30문제
    - 예준이만 빠지고, 시현, 민성, 대현 셋이 풀었을 경우 : 49문제

민성	시현	대현	예준	문제 개수
O	O	O	O	50
X	O	O	O	10
O	X	O	O	15
O	O	X	O	40
O	O	O	X	49



누가 가장 알고리즘을 푸는 데 가장 많이 기여했는가?

민성

Shapley Value

	feature1	feature2	feature3	예측 값
Case1	X	X	X	10
Case2	O	X	X	14
Case3	X	O	X	19
Case4	X	X	O	8
Case5	O	O	X	20
Case6	O	X	O	11
Case7	X	O	O	15
Case8	O	O	O	18

feature 1

② − ① = 14 − 10 ⇒ 4 × $\frac{1}{3}$ = $\frac{4}{3}$

⑤ − ③ = 20 − 19 ⇒ 1 × $\frac{1}{6}$ = $\frac{1}{6}$

⑥ − ④ = 11 − 8 ⇒ 3 × $\frac{1}{6}$ = $\frac{3}{6}$

⑧ − ⑦ = 18 − 15 ⇒ 3 × $\frac{1}{3}$ = 1

$\frac{4}{3} + \frac{1}{6} + \frac{3}{6} + 1 = 3$

└ shapley value

	feature1	feature2	feature3	예측 값
Case1	X	X	X	10
Case2	O	X	X	14
Case3	X	O	X	19
Case4	X	X	O	8
Case5	O	O	X	20
Case6	O	X	O	11
Case7	X	O	O	15
Case8	O	O	O	18

feature 2

③ − ① = 19 − 10 ⇒ 9 × $\frac{1}{3}$ = 3

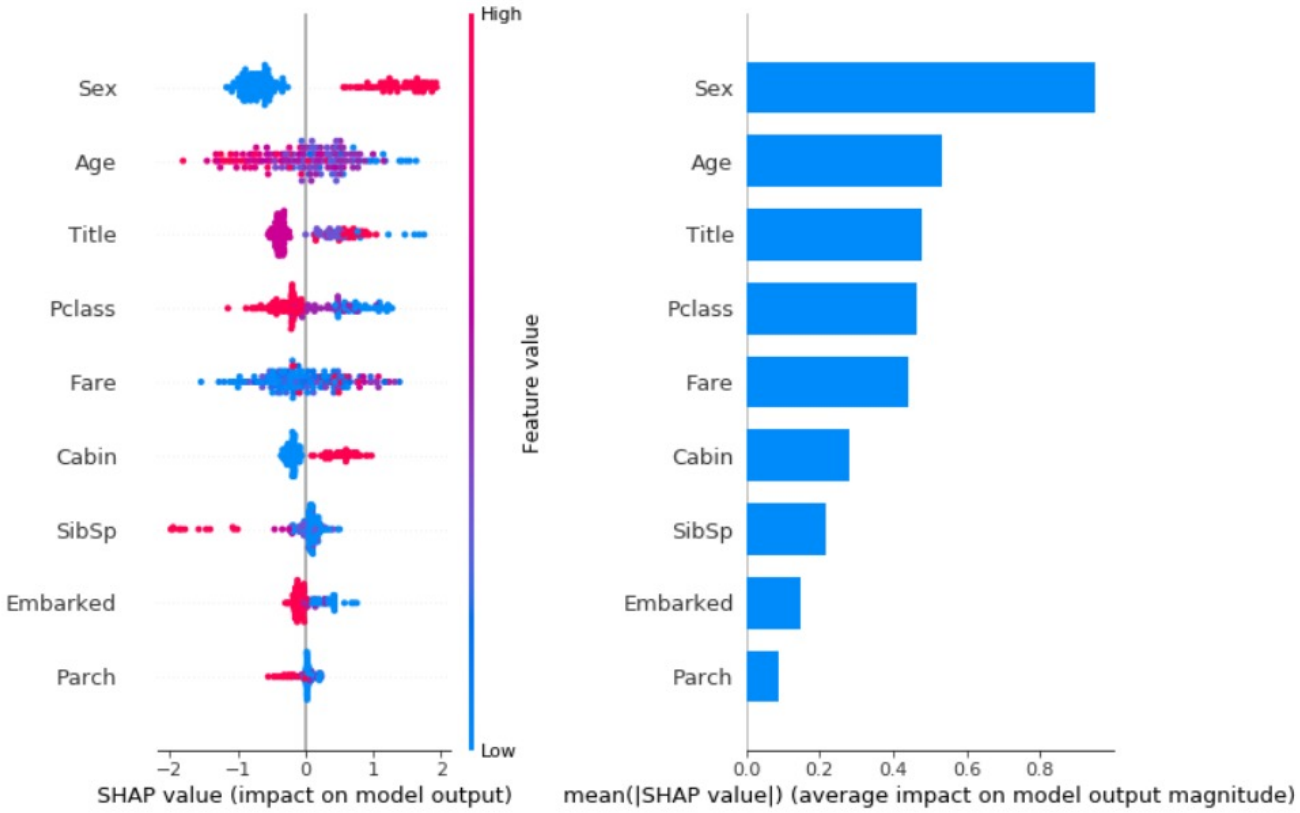
⑤ − ② = 20 − 14 = 6 × $\frac{1}{6}$ = 1

⑦ − ④ = 15 − 8 = 7 × $\frac{1}{6}$ = $\frac{7}{6}$

⑧ − ⑥ = 18 − 11 = 7 × $\frac{1}{3}$ = $\frac{7}{3}$ = $\frac{14}{6}$

$3 + 1 + \frac{7}{6} + \frac{14}{6} = \frac{45}{6} = 7.5$ xxx

SHAP value 그래프



타이타닉 사망 예측 데이터

색상 : 해당 변수의 값

음수일 수록 생존에 기여
양수일 수록 사망에 기여

남성 : 빨강색, 여성 : 파랑색

SHAP

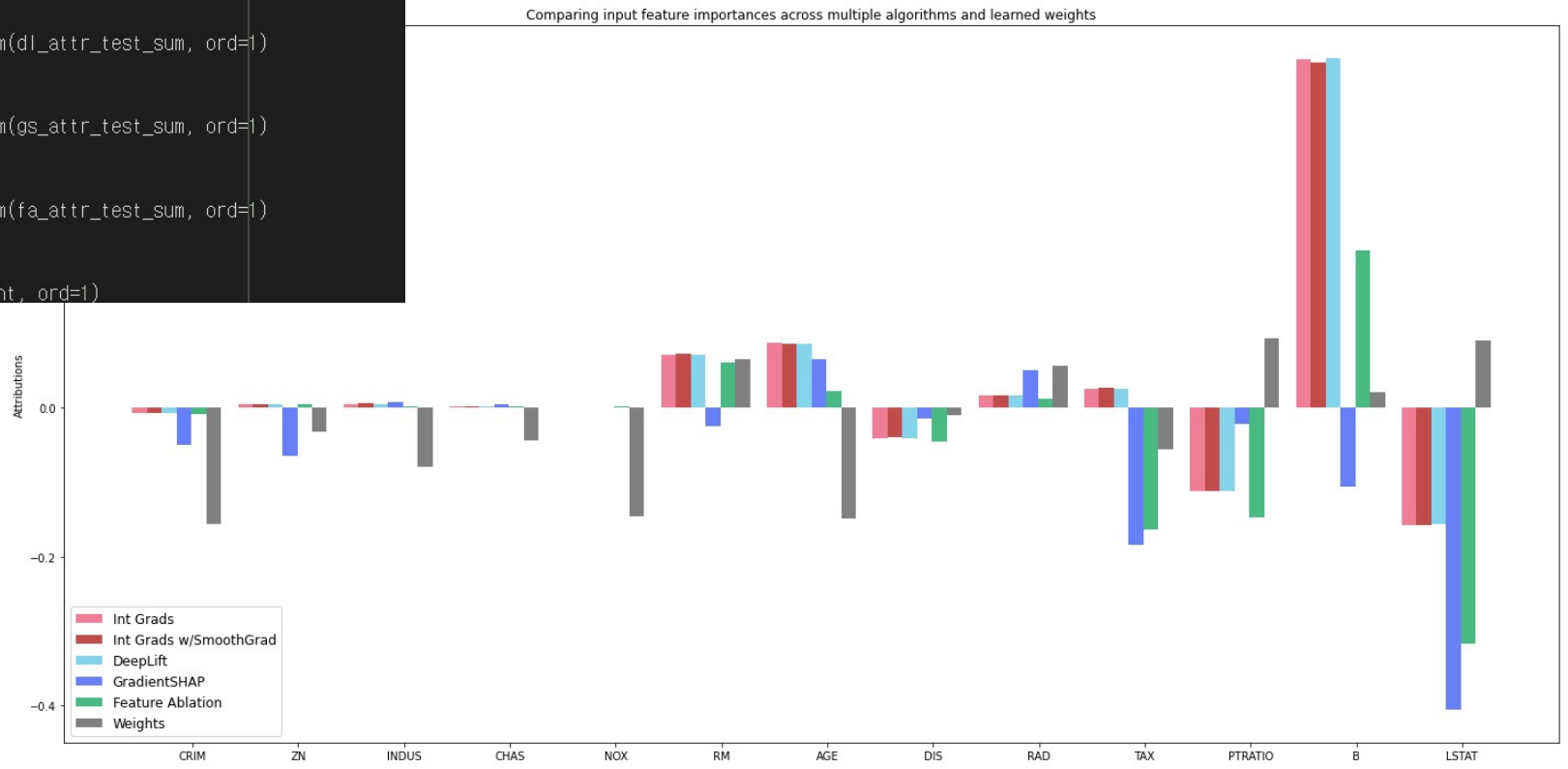
- Shapley Value의 문제점
  - 모든 순열 조합에 대해 연산하고 체크해야 하므로 연산 속도가 느림
- 모델의 특징에 따라 계산법을 변형시켜 연산 속도를 높임

1. Kernel SHAP: Linear LIME + Shapley Value
2. Tree SHAP: Tree based Model
3. Deep SHAP: DeepLearning based Model

코드 라이브러리

```
x_axis_data = np.arange(X_test.shape[1])
x_axis_data_labels = list(map(lambda idx: feature_names[idx], x_axis_data))

ig_attr_test_sum = ig_attr_test.detach().numpy().sum(0)
ig_attr_test_norm_sum = ig_attr_test_sum / np.linalg.norm(ig_attr_test_sum, ord=1)

ig_nt_attr_test_sum = ig_nt_attr_test.detach().numpy().sum(0)
ig_nt_attr_test_norm_sum = ig_nt_attr_test_sum / np.linalg.norm(ig_nt_attr_test_sum, ord=1)

dl_attr_test_sum = dl_attr_test.detach().numpy().sum(0)
dl_attr_test_norm_sum = dl_attr_test_sum / np.linalg.norm(dl_attr_test_sum, ord=1)

gs_attr_test_sum = gs_attr_test.detach().numpy().sum(0)
gs_attr_test_norm_sum = gs_attr_test_sum / np.linalg.norm(gs_attr_test_sum, ord=1)

fa_attr_test_sum = fa_attr_test.detach().numpy().sum(0)
fa_attr_test_norm_sum = fa_attr_test_sum / np.linalg.norm(fa_attr_test_sum, ord=1)

lin_weight = model.lin1.weight[0].detach().numpy()
y_axis_lin_weight = lin_weight / np.linalg.norm(lin_weight, ord=1)
```

```
# imports from captum library
from captum.attr import LayerConductance, LayerActivation, LayerIntegratedGradients
from captum.attr import IntegratedGradients, DeepLift, GradientShap, NoiseTunnel, FeatureAblation
from captum.attr import ShapleyValues
```



Q & A