Transformer-based Bayesian Inference for Tabular Data

2025.7.11

2024020757 조용수

jys1537@korea.ac.kr

발표자 소개

❖ **조용수 (Yongsu Jo)**
- 고려대학교 산업경영공학과 석사과정(2024.03 ~)
- Data Mining & Quality Analytics Labs. (김성범 교수님)

❖ **관심 연구 분야**
- Supervised Learning
- Tabular Data

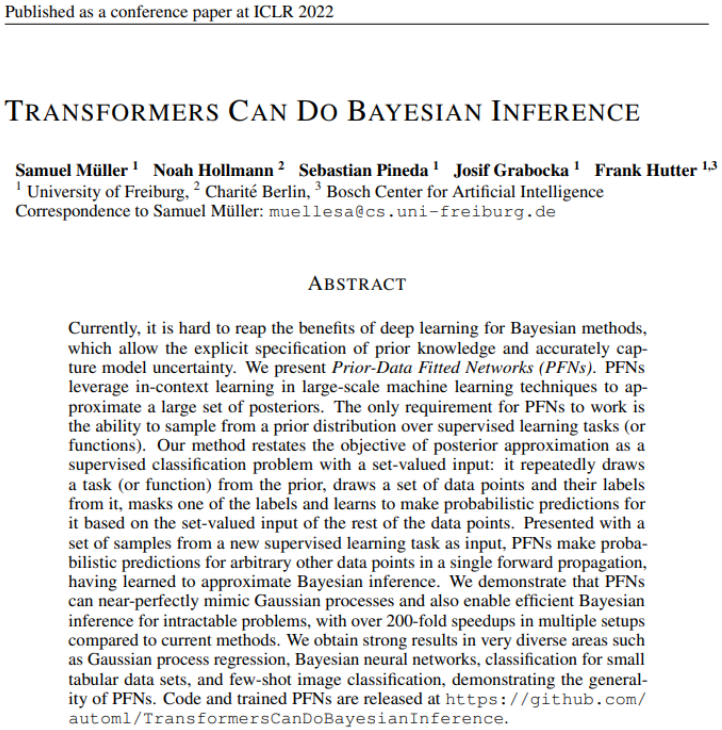
❖ **E-Mail**
- jys1537@korea.ac.kr

Contents

Transformer-Based Bayesian Inference for Tabular Data

❖ Transformer Can Do Bayesian Inference

- ICLR 2022 게재, 25년 7월 기준 224회 인용



Published as a conference paper at ICLR 2022

TRANSFORMERS CAN DO BAYESIAN INFERENCE

Samuel Müller [1] **Noah Hollmann** [2] **Sebastian Pineda** [1] **Josif Grabocka** [1] **Frank Hutter** [1,3]
[1] University of Freiburg, [2] Charité Berlin, [3] Bosch Center for Artificial Intelligence
Correspondence to Samuel Müller: muellesa@cs.uni-freiburg.de

ABSTRACT

Currently, it is hard to reap the benefits of deep learning for Bayesian methods, which allow the explicit specification of prior knowledge and accurately capture model uncertainty. We present *Prior-Data Fitted Networks (PFNs)*. PFNs leverage in-context learning in large-scale machine learning techniques to approximate a large set of posteriors. The only requirement for PFNs to work is the ability to sample from a prior distribution over supervised learning tasks (or functions). Our method restates the objective of posterior approximation as a supervised classification problem with a set-valued input: it repeatedly draws a task (or function) from the prior, draws a set of data points and their labels from it, masks one of the labels and learns to make probabilistic predictions for it based on the set-valued input of the rest of the data points. Presented with a set of samples from a new supervised learning task as input, PFNs make probabilistic predictions for arbitrary other data points in a single forward propagation, having learned to approximate Bayesian inference. We demonstrate that PFNs can near-perfectly mimic Gaussian processes and also enable efficient Bayesian inference for intractable problems, with over 200-fold speedups in multiple setups compared to current methods. We obtain strong results in very diverse areas such as Gaussian process regression, Bayesian neural networks, classification for small tabular data sets, and few-shot image classification, demonstrating the generality of PFNs. Code and trained PFNs are released at https://github.com/automl/TransformersCanDoBayesianInference.

Introduction

Tabular Data

❖ Tabular Data
- 행 (Row, Instance)과 열(Column, Feature)로 표현되는 데이터

특징

관측치

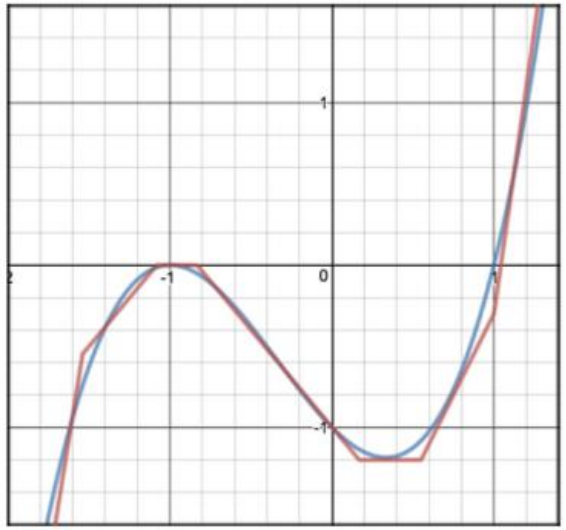
학생 이름	과목	점수	마법 지팡이 재료	최애 주문
해리 포터	어둠의 방어술	95	호랑가시나무	익스펙토 페트로눔
헤르미온느 그레인저	변신술	100	포도나무	레비오사
론 위즐리	마법약	72	회양목	레둑토
말포이	마법 역사	50	호손	인센디오
네빌 롱바텀	식물학	85	체리나무	알로호모라

관계

Background

Universal Approximation Theorem

Universal Approximation Theorem



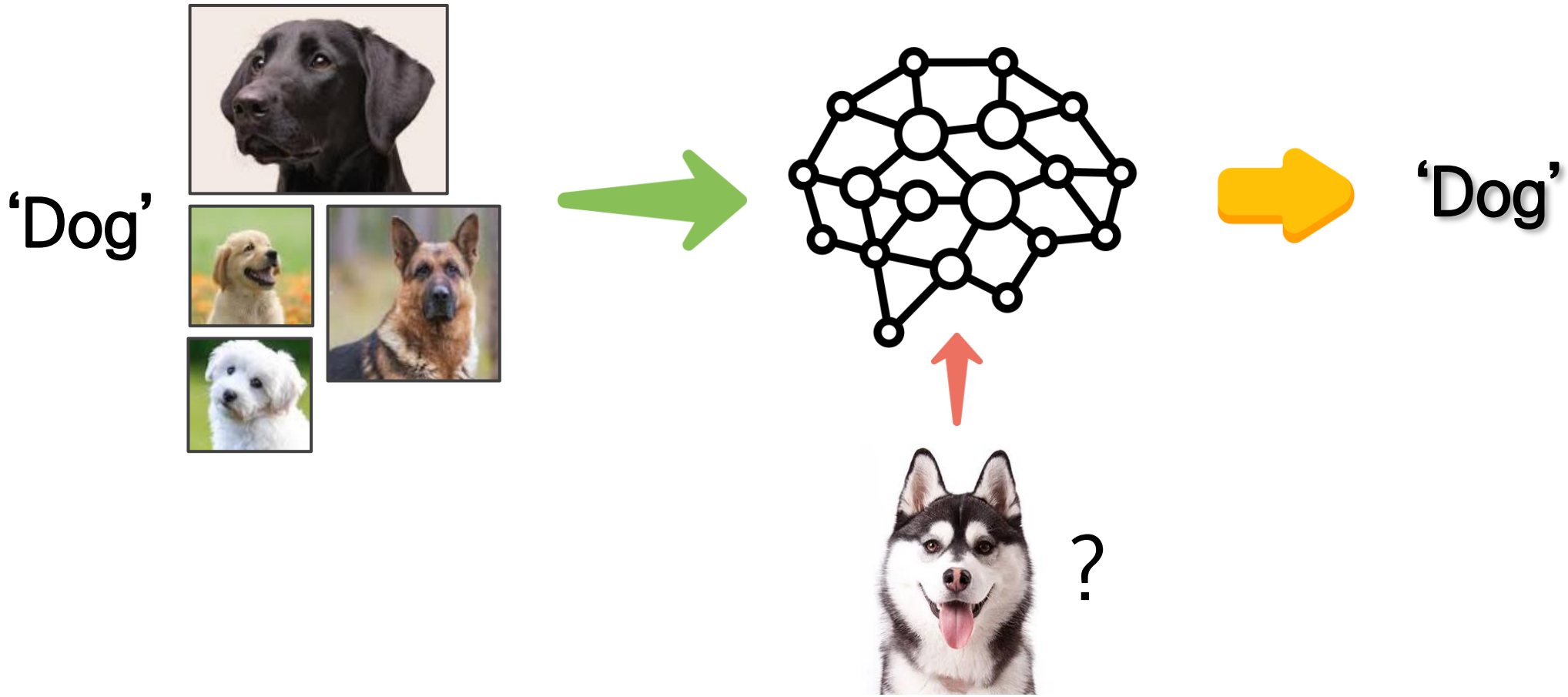
$$n_1(x) = Relu(-5x - 7.7)$$
$$n_2(x) = Relu(-1.2x - 1.3)$$
$$n_3(x) = Relu(1.2x + 1)$$
$$n_4(x) = Relu(1.2x - .2)$$
$$n_5(x) = Relu(2x - 1.1)$$
$$n_6(x) = Relu(5x - 5)$$

$$Z(x) = -n_1(x) - n_2(x) - n_3(x) + n_4(x) + n_5(x) + n_6(x)$$

“하나 이상의 은닉층과 비선형 활성함수를 갖는 신경망은 충분한 뉴런의 수만 있다면 임의의 연속함수를 임의의 오차 이하로 근사할 수 있다.”

Background

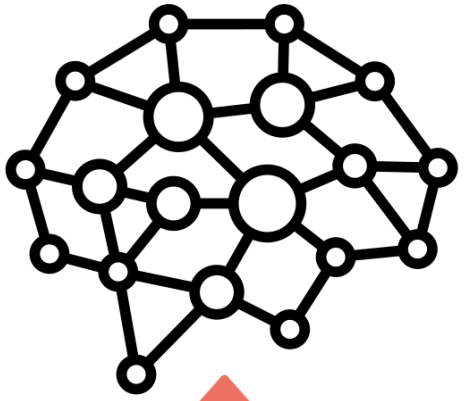
Supervised Learning



Background

Supervised Learning

‘Dog’



‘Dog’

?

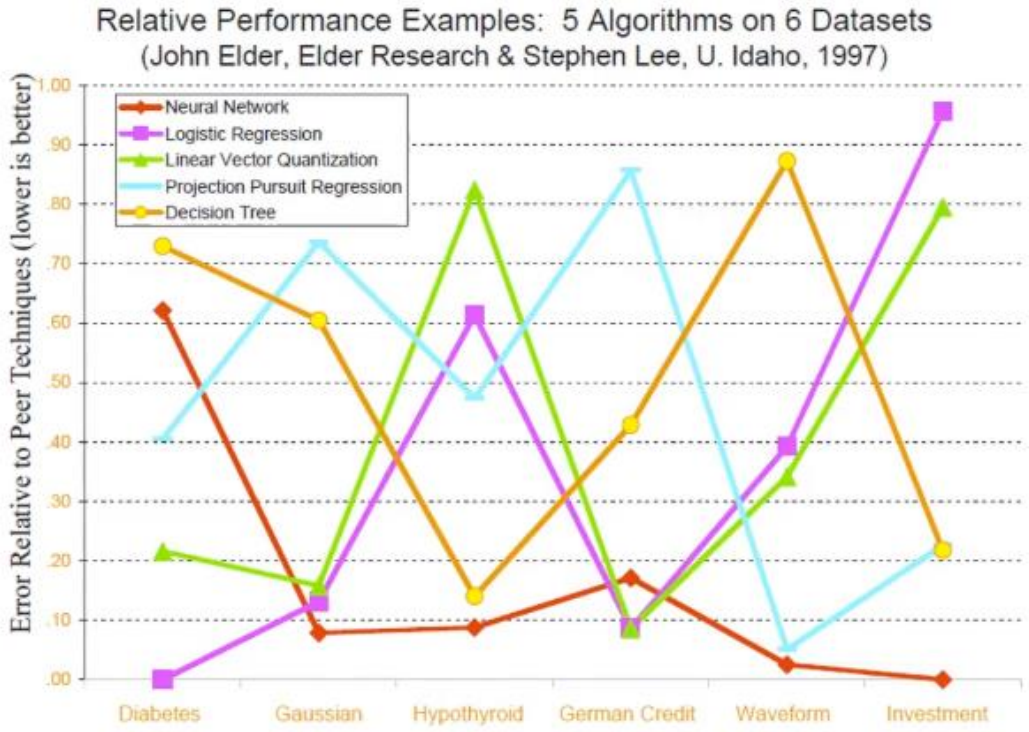
Background

No Free Lunch Theorem

IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, VOL. 1, NO. 1, APRIL 1997 67

No Free Lunch Theorems for Optimization

David H. Wolpert and William G. Macready

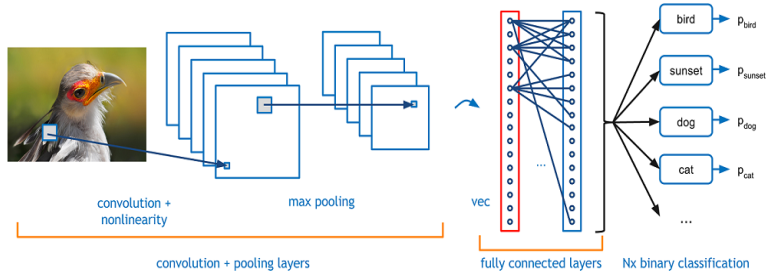


"모든 문제에 대해 항상 우수한 성능을 내는 단일 알고리즘은 존재하지 않는다"

Background

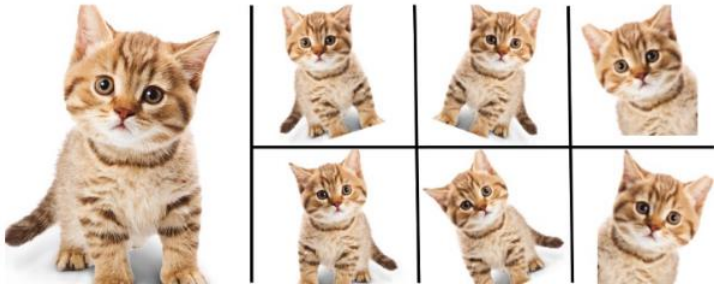
Prior Knowledge

CNN



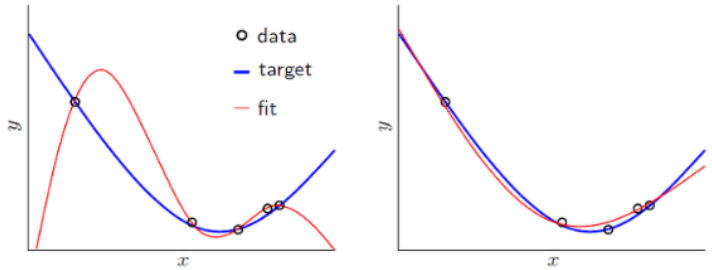
"이미지에 Local Pattern 이 있다."

Data Augmentation



"회전/크기/위치가 달라도 같은 Class일 것이다."

Regularization



"모델 파라미터가 적을수록 더 좋은 일반화를 한다."

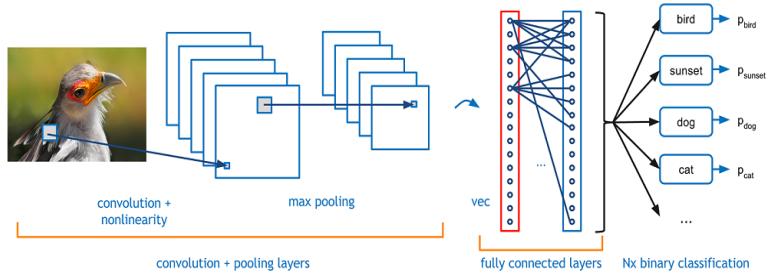
Background

Prior Knowledge

Inductive Bias (귀납적 편향)

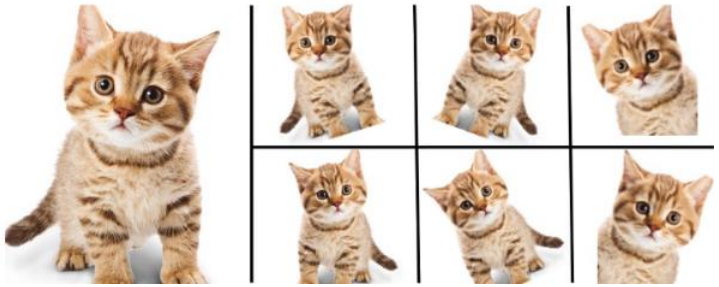
Prior Knowledge

CNN



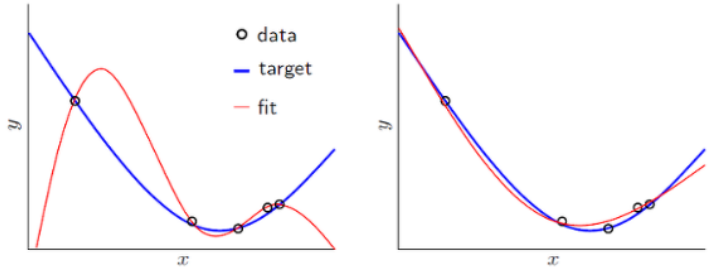
"이미지에 Local Pattern 이 있다."

Data Augmentation



"회전/크기/위치가 달라도 같은 Class일 것이다."

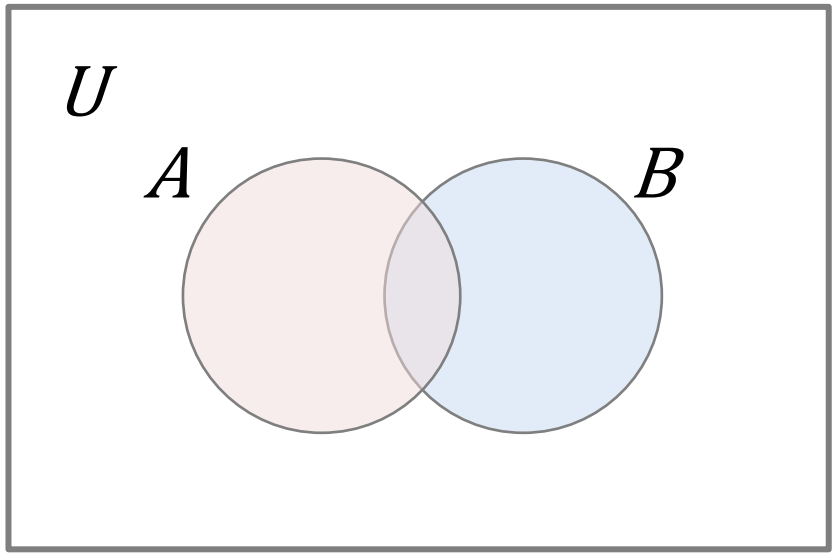
Regularization



"모델 파라미터가 적을수록 더 좋은 일반화를 한다."

Background

Bayesian Theorem



사건 A가 일어날 확률 : $P(A)$ / 사건 B가 일어날 확률 : $P(B)$

사건 A가 일어난 상태에서 사건 B가 일어난 확률 : $P(A|B) = \frac{P(A \cap B)}{P(B)}$

사건 B가 일어난 상태에서 사건 A가 일어난 확률 : $P(B|A) = \frac{P(A \cap B)}{P(A)}$

$$P(A \cap B) = P(A|B) \cdot P(B) = P(B|A) \cdot P(A)$$

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)}$$

Background

Frequentism vs Bayesianism

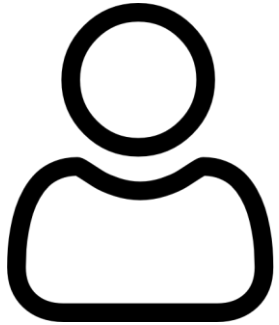
동전을 던져서 앞면이 나올 확률은 50%이다

Background

Frequentism vs Bayesianism

동전을 던져서 앞면이 나올 확률은 50%이다

Frequentism : 동전을 100번 던지면 앞면은 50번 나올 것!



Bayesianism : 동전을 던졌을 때 앞면이 나온다고 50% 판단할 수 있어

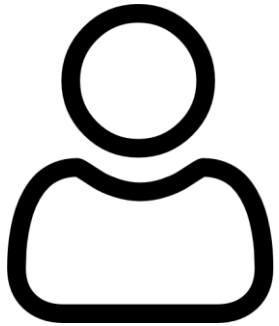
Background

Frequentism vs Bayesianism

동전을 던져서 앞면이 나올 확률은 50%이다

Frequentism : 동전을 100번 던지면 앞면은 50번 나올 것!

→ 사건이 일어나는 장기적인 확률로 경험적인 사실에 기반



Bayesianism : 동전을 던졌을 때 앞면이 나온다고 50% 판단할 수 있어

→ 지식이나 판단의 정도를 나타내는 수단

Background

Bayesian Theorem

$$\underbrace{P(y|x,D)}_{\text{사후예측분포}} = \int \underbrace{P(y|x,\theta)}_{\text{Likelihood}} \overbrace{P(\theta|D)}^{\text{사후분포}} d\theta$$

PPD : Posterior Predictive Distribution

조건부 예측 분포

$$P(\theta|D) = \frac{P(D|\theta)\cdot P(\theta)}{P(D)}$$

θ : 모델의 파라미터

D : 관측된 데이터

$P(\theta)$: Prior – 사전확률, Prior (우리가 가진 Prior Belief, 귀납적 편향)

$P(\theta|D)$: Posterior – 데이터가 주어졌을 때 확률 – 사후 분포, Posterior (데이터 관측 후 θ 가 참일 확률 분포)

$P(D|\theta)$: Likelihood – 데이터를 관측한 후의 파라미터에 대한 확률 (주어진 θ에서 데이터가 얼마나 설명되는가)

$P(D)$: Evidence – 가능한 모든 θ에 대해 Marginalization $P(D) = \int P(D|\theta)\cdot P(\theta)d\theta$

*Marginalization : 어떤 변수에 대해 전체 확률 계산하기 위해 다른 변수들을 적분하여 제거하는 과정

Background

Bayesian Theorem

$$P(covid|\text{기침}) = \frac{P(\text{기침}|covid)\cdot P(covid)}{P(\text{기침})}$$

누군가 기침을 하고 있다. 가능한 원인은 감기 or Covid

우리는 이 사람이 코로나일 확률을 구하고자 한다. $\Rightarrow P(covid|\text{기침})$

$P(covid)$: Prior – 사전확률 (사람이 covid 걸릴 사전확률 – 전체 인구 중 covid 감염률)

$P(covid|\text{ 기침 })$: Posterior – 사후 분포 (기침하는 사람 중 코로나일 확률, 구하고자 하는 것)

$P(\text{기침 }|covid)$: Likelihood – 코로나 환자가 기침 할 확률

$P(D)$: Evidence – 누군가 기침을 할 확률

Background

Bayesian Theorem

$$P(covid|\text{기침}) = \frac{P(\text{기침}|covid)\cdot P(covid)}{P(\text{기침})}$$

누군가 기침을 하고 있다. 가능한 원인은 감기 or Covid

우리는 이 사람이 코로나일 확률을 구하고자 한다. $\Rightarrow P(covid|\text{기침})$

$P(covid)$: Prior – 사전확률 (사람이 covid 걸릴 사전확률 – 전체 인구 중 covid 감염률) **Fixed**

$P(covid|\text{ 기침 })$: Posterior – 사후 분포 (기침하는 사람 중 코로나일 확률, 구하고자 하는 것)

$P(\text{기침 }|covid)$: Likelihood – 코로나 환자가 기침 할 확률 **계산 가능**

$P(D)$: Evidence – 누군가 기침을 할 확률

$P(D) = \int P(D|\theta)\cdot P(\theta)d\theta$ → 실제 많은 문제에서 복잡한 확률의 계산 - **Intractability**

Background

Bayesian Theorem

$$P(covid|\text{기침}) = \frac{P(\text{기침}|covid)\cdot P(covid)}{P(\text{기침})}$$

누군가 기침을 하고 있다. 가능한 원인은 감기 or Covid

우리는 이 사람이 코로나일 확률을 구하고자 한다. $\Rightarrow P(covid|\text{기침})$

$P(covid)$: Prior – 사전확률 (사람이 covid 걸릴 사전확률 – 전체 인구 중 covid 감염률) **Fixed**

$P(covid|\text{ 기침 })$: Posterior – 사후 분포 (기침하는 사람 중 코로나일 확률, 구하고자 하는 것) **계산 어려움**

$P(\text{기침 }|covid)$: Likelihood – 코로나 환자가 기침 할 확률 **계산 가능**

$P(D)$: Evidence – 누군가 기침을 할 확률 **계산 어려움**

$P(D) = \int P(D|\theta)\cdot P(\theta)d\theta$ → 실제 많은 문제에서 복잡한 확률의 계산 - **Intractability**

Background

Bayesian Theorem

이렇게 복잡하고 어려운 방식의 Bayesian 추론을 해야하나?

장점 : 1. 이론적 타당성 (Theoretical Soundness)

사전 확률 p(t) 가 실제 데이터 생성 과정을 적절히 반영할 경우, 베이지안 추론은 강력한 이론적 기반을 갖는다.

베이지안 추론은 확률을 통해 불확실성을 다루는 가장 정합(Consistent)한 방법

Background

Bayesian Theorem

이렇게 복잡하고 어려운 방식의 Bayesian 추론을 해야하나?

장점 : 1. 이론적 타당성 (Theoretical Soundness)

2. 불확실성의 정교한 반영 (Quantified Uncertainty)

단순한 예측값이 아닌 분포를 제공

Background

Bayesian Theorem

이렇게 복잡하고 어려운 방식의 Bayesian 추론을 해야하나?

장점 : 1. 이론적 타당성 (Theoretical Soundness)

2. 불확실성의 정교한 반영 (Quantified Uncertainty)

3. 예측의 보정성 (Well-Calibrated Predictions)

실제 확률과 일치하는 예측을 제공

Background

Bayesian Theorem

이렇게 복잡하고 어려운 방식의 Bayesian 추론을 해야하나?

장점 : 1. 이론적 타당성 (Theoretical Soundness)

2. 불확실성의 정교한 반영 (Quantified Uncertainty)

3. 예측의 보정성 (Well-Calibrated Predictions)

4. 모델 해석 가능성 (Interpretability via Prior)

모델에 입력한 사전 지식을 수학적으로 명시하여 해석에 용이

Background

Bayesian Theorem

이렇게 복잡하고 어려운 방식의 Bayesian 추론을 해야하나?

장점 : 1. 이론적 타당성 (Theoretical Soundness)

2. 불확실성의 정교한 반영 (Quantified Uncertainty)
3. 예측의 보정성 (Well-Calibrated Predictions)
4. 모델 해석 가능성 (Interpretability via Prior)

VI? Laplace?

MCMC?

$$P(\theta|D) = \frac{P(D|\theta)\cdot P(\theta)}{P(D)}$$

$P(D) = \int P(D|\theta) \cdot P(\theta) d\theta$ → 실제 많은 문제에서 복잡한 확률의 계산 - **Intractability**

Prior-Data Fitted Network

$$\underbrace{P(y|x,D)}_{\text{사후예측분포}} = \int \underbrace{P(y|x,\theta)}_{\substack{\text{Likelihood} \\ \text{조건부 예측 분포}}} \overbrace{P(\theta|D)}^{\text{사후분포 (Posterior)}} d\theta$$

가능한 모든 θ 하의 **조건부 예측 분포**에 따라 이들의 **가능성(사후분포)**에 따른 가중 평균

➔ 사후 분포의 계산이 어려워서 MCMC/VI 등의 근사를 사용하지만 Cost가 크고 복잡함

Prior-Data Fitted Network

$$\underbrace{P(y|x,D)}_{\text{사후예측분포}} = \int \underbrace{P(y|x,\theta)}_{\substack{\text{Likelihood} \\ \text{조건부 예측 분포}}} \overbrace{P(\theta|D)}^{\text{사후분포 (Posterior)}} d\theta$$

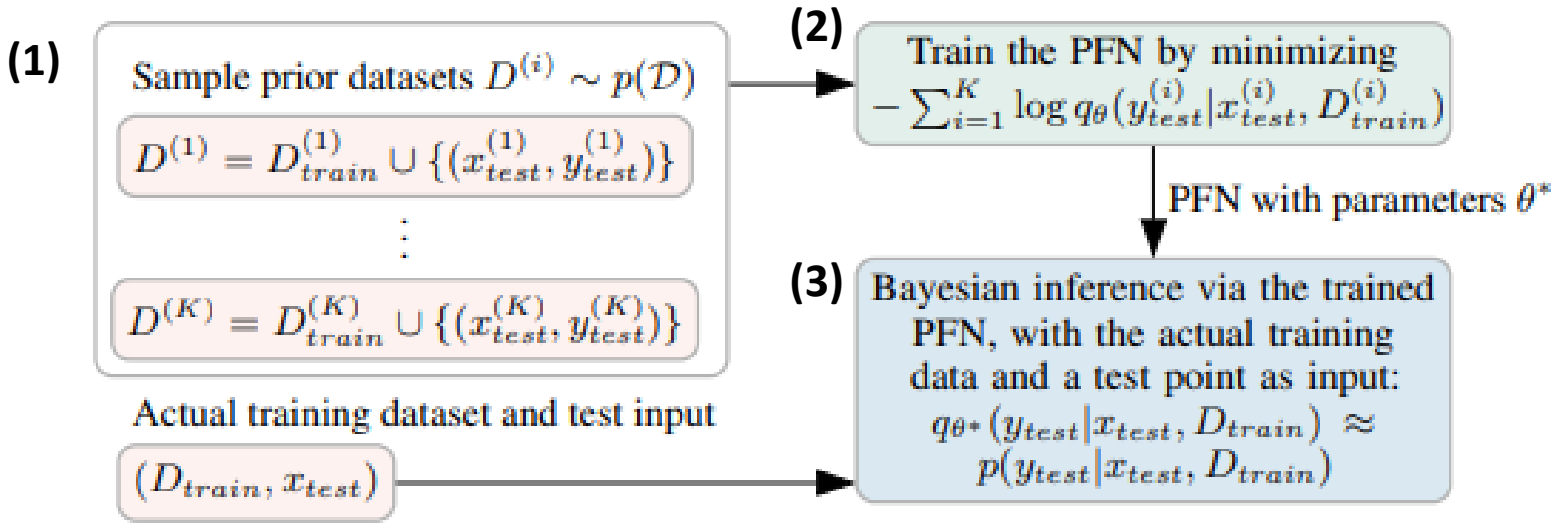
가능한 모든 θ 하의 **조건부 예측 분포**에 따라 이들의 **가능성(사후분포)**에 따른 가중 평균

➔ 사후 분포의 계산이 어려워서 MCMC/VI 등의 근사를 사용하지만 Cost가 크고 복잡함

어차피 우리가 필요한 건 θ 가 아니라 $P(y|x,D)$인데
혹시 모델로 직접 학습 할 수 있지 않을까?

Prior-Data Fitted Network

- PFN (Prior-Data Fitted Network)의 학습 및 추론 과정
  - (1) Prior에서 데이터셋 샘플링 : 우리가 정의한 사전 분포(Prior Distribution, $p(D)$)로 부터 여러 데이터 셋 D 샘플링
  - (2) PFN 학습 : Transformer기반 네트워크로 한 개의 데이터를 정답, 나머지 데이터를 입력 D로 학습을 반복함
  - (3) 실제 추론 : 새로운 데이터 셋 D와 test 입력 x가 주어지면, 한번의 Forward pass 계산으로 PPD 출력



Prior-Data Fitted Network

- PFN (Prior-Data Fitted Network)의 학습 및 추론 과정

  (1) Prior에서 데이터셋 샘플링 : 우리가 정의한 사전 분포(Prior Distribution, $p(D)$)로 부터 여러 데이터 셋 D 샘플링

1. GP 기반 Prior Dataset 생성

1) 입력 샘플링

- $x_i \sim \text{Uniform}[0,1]^d$ 형태로, 입력 포인트 $x_1, \ldots, x_N$를 균등 샘플링

2) 커널 기반 공분산 계산

- 특정 커널 함수 $k(x_i, x_j)$ (e.g,. RBF kernel)을 사용해 공분산 행렬 K생성

$$K_{i,j} = k(x_i, x_j)$$

3) 다변량 정규분포로부터 출력 y 샘플링

- 평균 0, 공분산 K를 가지는 정규분포 $N(0, K)$에서 출력 벡터 y 샘플링

$$y \sim N(0, K)$$

(1)

Sample prior datasets $D^{(i)} \sim p(\mathcal{D})$

$$D^{(1)} = D^{(1)}_{train} \cup \{(x^{(1)}_{test}, y^{(1)}_{test})\}$$

$$\vdots$$

$$D^{(K)} = D^{(K)}_{train} \cup \{(x^{(K)}_{test}, y^{(K)}_{test})\}$$

Actual training dataset and test input

Prior-Data Fitted Network

- PFN (Prior-Data Fitted Network)의 학습 및 추론 과정

  (1) Prior에서 데이터셋 샘플링 : 우리가 정의한 사전 분포(Prior Distribution, $p(D)$)로 부터 여러 데이터 셋 D 샘플링

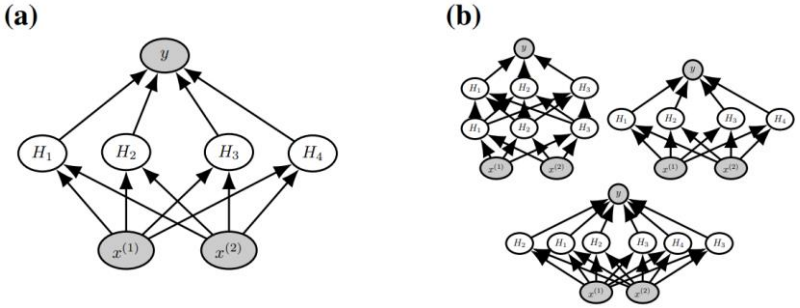
2. BNN 기반 Prior Dataset 생성

1) 모델 아키텍쳐 A 샘플링 : $A \sim \mathrm{p}(A)$

2) 해당 아키텍쳐 가중치 $W_{i,j}$샘플링 : $W_{i,j} \sim p_w(\cdot)$

3) 각 데이터 포인트 x_i의 feature $x_{i,f}$를 $N(0, K)$에서 샘플링

4) 샘플링된 아키텍쳐 A에 x_i를 통과시켜 $y_i = Aw(x_i)$ 생성



BNN

A prior over
BNN architectures

(1)

Sample prior datasets $D^{(i)} \sim p(\mathcal{D})$

$$D^{(1)} = D^{(1)}_{train} \cup \{(x^{(1)}_{test}, y^{(1)}_{test})\}$$

$$\vdots$$

$$D^{(K)} = D^{(K)}_{train} \cup \{(x^{(K)}_{test}, y^{(K)}_{test})\}$$

Actual training dataset and test input

Prior-Data Fitted Network

- PFN (Prior-Data Fitted Network)의 학습 및 추론 과정

  (2) PFN 학습 : Transformer기반 네트워크로 한 개의 데이터를 정답, 나머지 데이터를 입력 D로 학습을 반복함

PFN q_θ를 D와 prior $\mathrm{p}(D)$에서 생성한 $\{(x_i, y_i)\}_{i=1}^{m}$ 로 학습

(2) Train the PFN by minimizing $-\sum_{i=1}^{K} \log q_\theta(y_{test}^{(i)} | x_{test}^{(i)}, D_{train}^{(i)})$

Loss function : Cross Entropy

$$\ell_\theta = -\mathbb{E}_{(x,y)\sim p_{\text{data}}}[\log q_\theta(y \mid x)]$$ 전통적인 Supervised Learning 의 Negative log-likelihood

$$\ell_\theta = -\mathbb{E}_{(x,y,D)\sim p(D)}[\log q_\theta(y \mid x, D)]$$

$$= -\int p(x, y, D) \log q_\theta(y \mid x, D)$$

$$= -\int p(x, D) \left(\int p(y \mid x, D) \log q_\theta(y \mid x, D)\, dy \right)$$

$$= \int p(x, D) \cdot H(p(\cdot \mid x, D), q_\theta(\cdot \mid x, D))$$

$$= \mathbb{E}_{x,D\sim p(D)}[\underbrace{H(p(\cdot \mid x, D)}_{\text{PPD}}, \underbrace{q_\theta(\cdot \mid x, D))}_{\text{예측분포}}]$$ ← Cross Entropy (PFN 의 예측분포 q_θ & PPD p)

Prior-Data Fitted Network

- PFN (Prior-Data Fitted Network)의 학습 및 추론 과정

  (3) 실제 추론 : 새로운 데이터 셋 D와 test 입력 x가 주어지면, 한번의 Forward pass 계산으로 PPD 출력

1. 추론 진행을 위한 구조는 트랜스포머 인코더 기반

   1) Permutation Invariant 구조

   데이터는 무작위 Sampling을 진행하므로 Positional Encoding 없이 설계하여 순서에 영향 받지 않도록 함

   2) 입력 구성

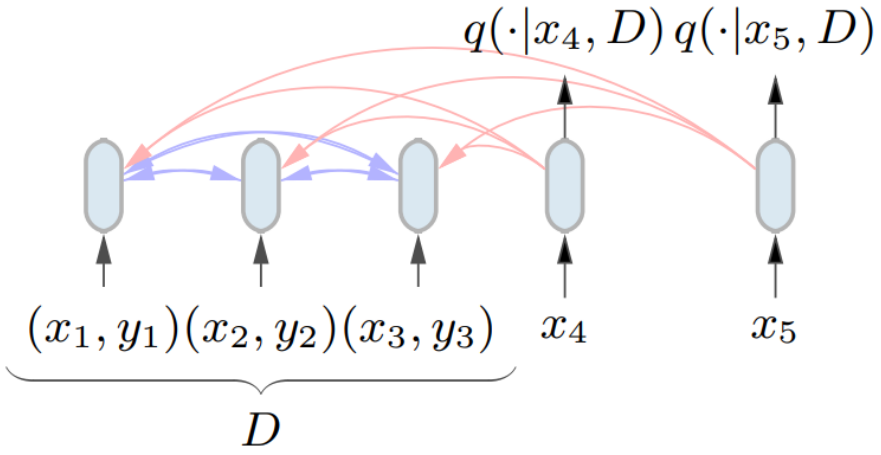
   N=n+m 개의 입력을 받음

   n개의 Training set, m개의 Query point

   Query Point에는 masking 하여 y 참조하지 않도록

(3)

Bayesian inference via the trained PFN, with the actual training data and a test point as input:
$q_{\theta^*}(y_{test}|x_{test}, D_{train}) \approx p(y_{test}|x_{test}, D_{train})$



Prior-Data Fitted Network

- PFN (Prior-Data Fitted Network)의 학습 및 추론 과정

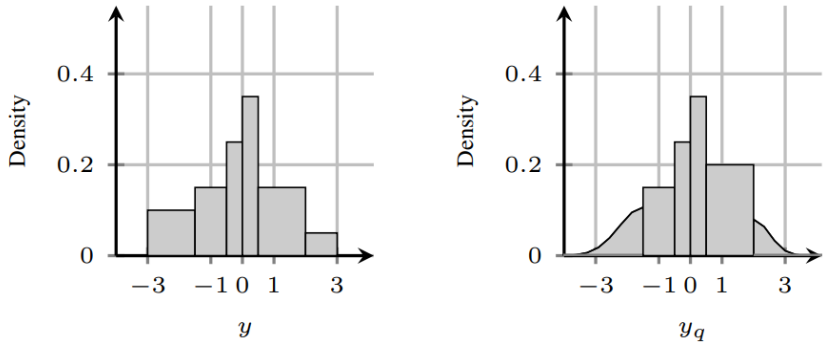
  (3) 실제 추론 : 새로운 데이터 셋 D와 test 입력 x가 주어지면, 한번의 Forward pass 계산으로 PPD 출력

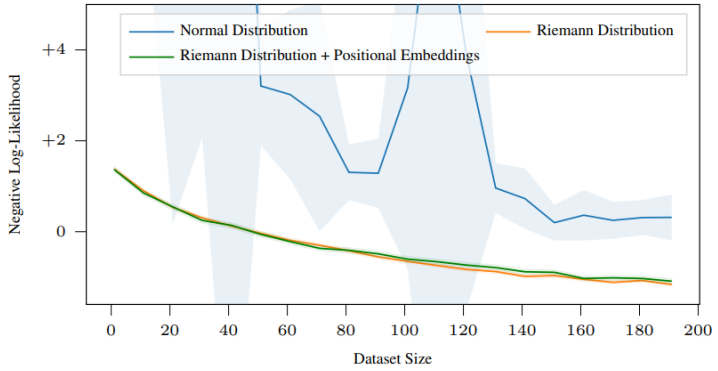
2. 연속적 확률 분포를 예측하기 위해 Riemann Distribution 도입

   -. 문제점 : Neural network는 Classification에는 잘 작동하지만,
   정밀한 연속 확률 분포(PPD) 예측에는 어려움이 있음.

(3) Bayesian inference via the trained PFN, with the actual training data and a test point as input: $q_{\theta^*}(y_{test}|x_{test}, D_{train}) \approx p(y_{test}|x_{test}, D_{train})$

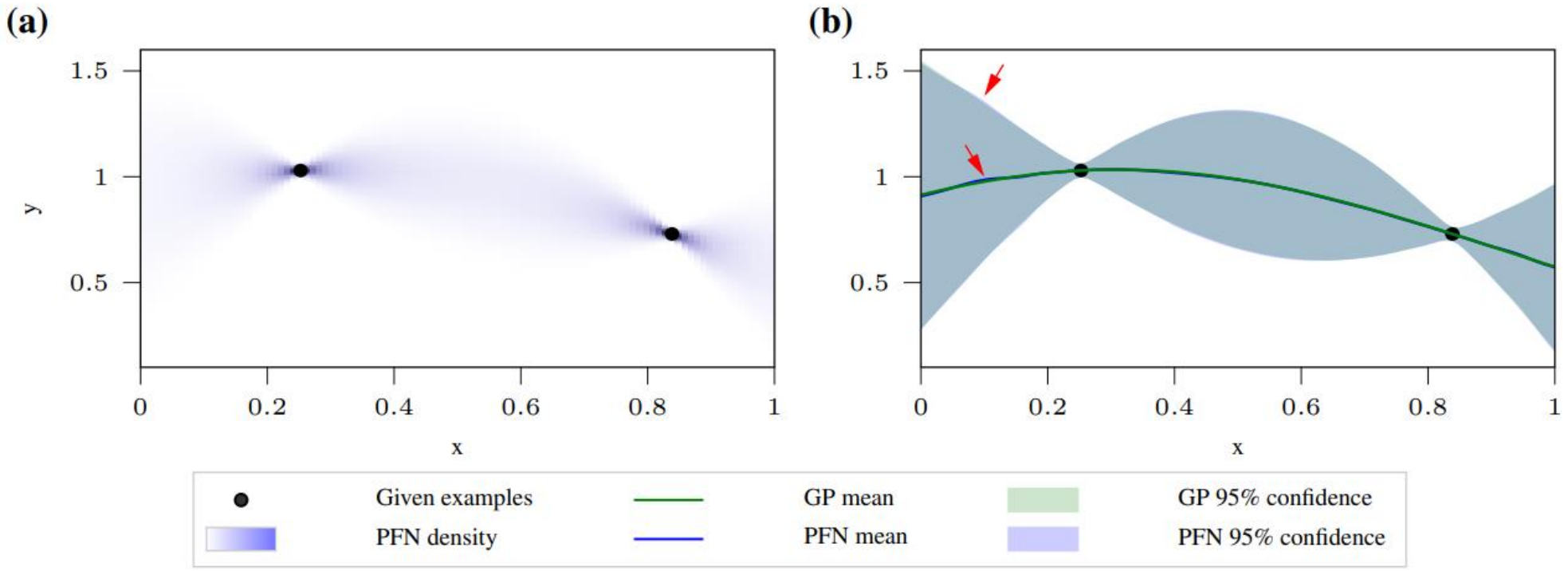
distributional reinforcement learning에서의 **discretized distribution** 개념에서 착안





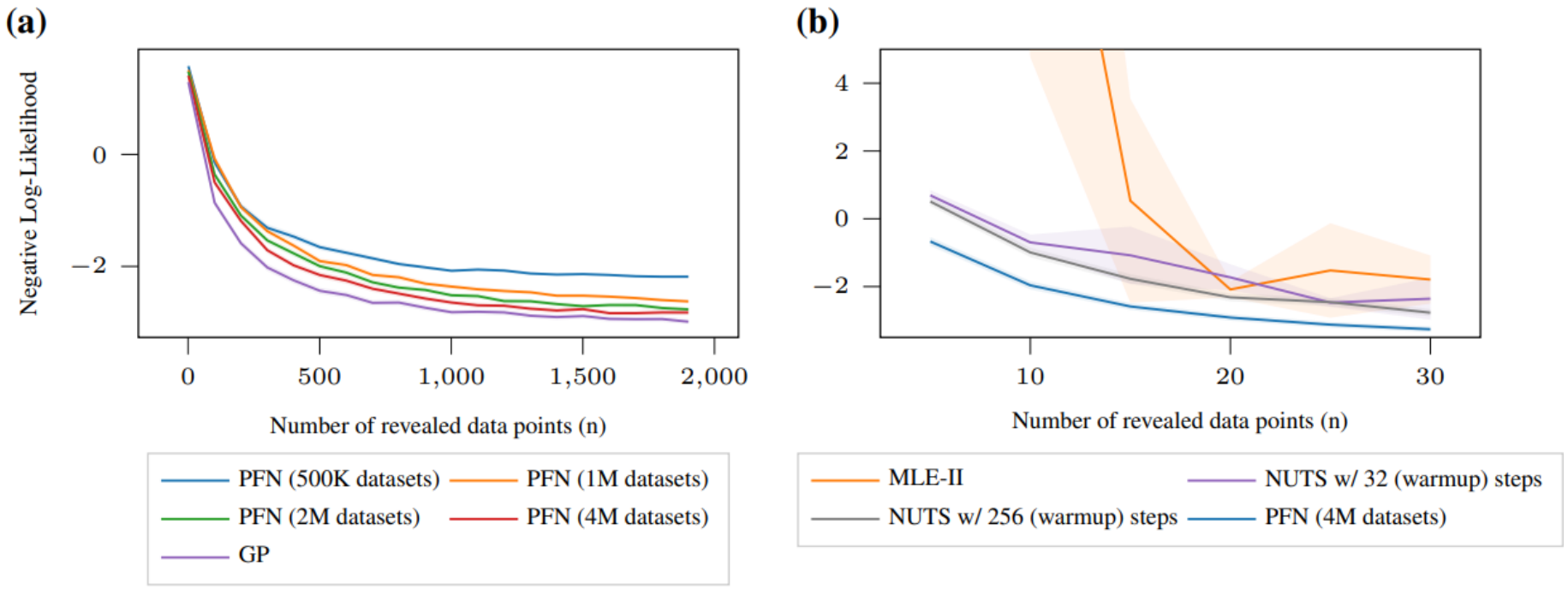
Experiments

- Gaussian Process 방식에서 2개의 관측값을 입력했을 때 PFN과 Bayesian 방식으로 계산된 PPD의 비교



→ PFN 방식이 별도의 사후 확률 계산 없이도 매우 유사하게 근사할 수 있음

Experiments



(a) Closed form Gaussian Process의 NLL과 PFN dataset 개수에 따른 NLL 비교
(b) Hyperparameter 를 변경하며 Gaussian Process로 생성한 데이터에 대한 Inference 방법별 NLL 비교

Transformer-Based Bayesian Inference for Tabular Data

❖ **TABPFN: A Transformer that solves small tabular classification problems in a seconds**

- ICLR 2023 게재, 25년 7월 기준 409회 인용

Published as a conference paper at ICLR 2023

TABPFN: A TRANSFORMER THAT SOLVES SMALL TABULAR CLASSIFICATION PROBLEMS IN A SECOND

Noah Hollmann[*,1,2] **Samuel Müller**[*,1] **Katharina Eggensperger**[1] **Frank Hutter**[1,3]
[1] University of Freiburg, [2] Charité University Medicine Berlin
[3] Bosch Center for Artificial Intelligence [*] Equal contribution.
Correspondence to noah.hollmann@charite.de & muellesa@cs.uni-freiburg.de

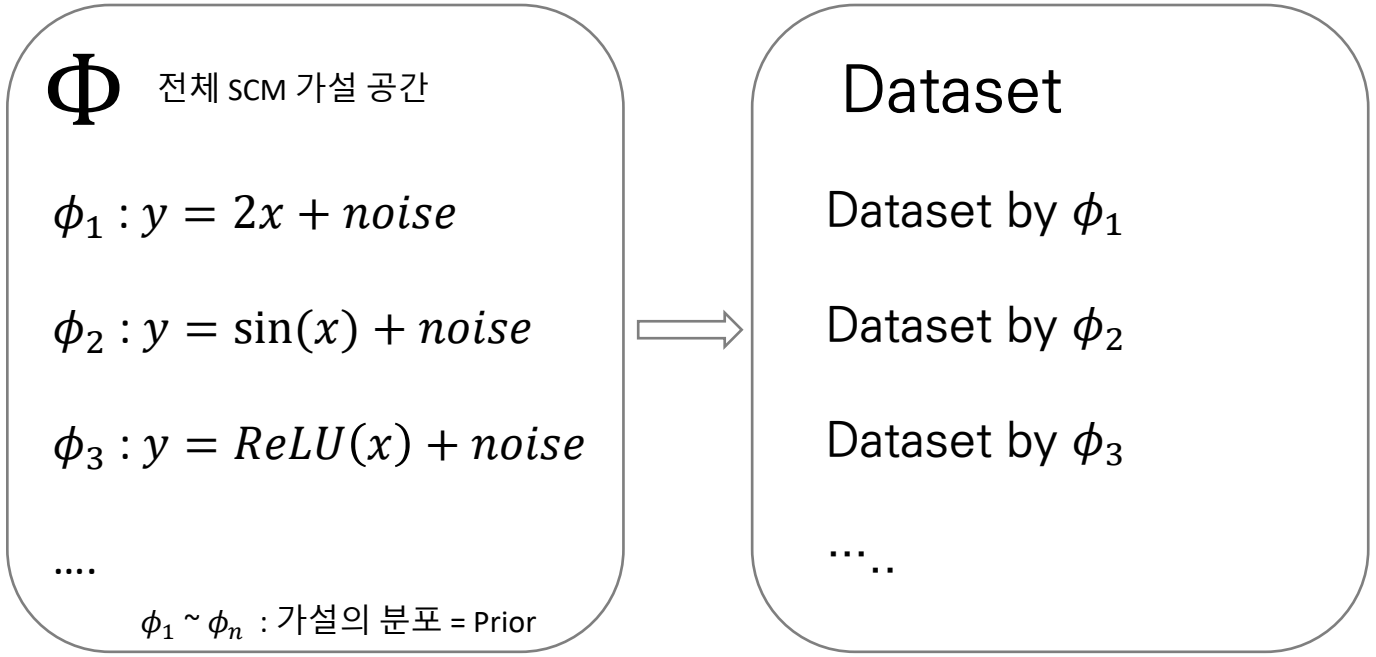
ABSTRACT

We present TabPFN, a trained Transformer that can do supervised classification for small tabular datasets in *less than a second*, needs no hyperparameter tuning and is competitive with state-of-the-art classification methods. TabPFN performs in-context learning (ICL), it learns to make predictions using sequences of labeled examples (x, f(x)) given in the input, without requiring further parameter updates. TabPFN is fully entailed in the weights of our network, which accepts training and test samples as a set-valued input and yields predictions for the entire test set in a single forward pass. TabPFN is a Prior-Data Fitted Network (PFN) and is trained offline once, to approximate Bayesian inference on synthetic datasets drawn from our prior. This prior incorporates ideas from causal reasoning: It entails a large space of structural causal models with a preference for simple structures. On the 18 datasets in the OpenML-CC18 suite that contain up to 1 000 training data points, up to 100 purely numerical features without missing values, and up to 10 classes, we show that our method clearly outperforms boosted trees and performs on par with complex state-of-the-art AutoML systems with up to 230× speedup. This increases to a 5 700× speedup when using a GPU. We also validate these results on an additional 67 small numerical datasets from OpenML. We provide all our code, the trained TabPFN, an interactive browser demo and a Colab notebook at https://github.com/automl/TabPFN.

Background

The Posterior Predictive Distribution for Supervised Learning

$$p(y \mid x, D) \propto \int_{\Phi} p(y \mid x, \phi)\, p(D \mid \phi)\, p(\phi)\, d\phi$$



Background

Synthetic Prior-Fitting

$$p(D) = \mathbb{E}_{\phi \sim p(\phi)}[p(D \mid \phi)]$$

Dataset

Dataset by ϕ_1

Dataset by ϕ_2

Dataset by ϕ_3

….

$\{x_1, y_1\}$
$\{x_2, y_2\}$
$\{x_3, y_3\}$
…
$\{x_{test}, y_{test}\} \leftarrow test\ point$
masking

Transformer 학습
(Parameter Θ 최적화)

다양한 가설을 통해 생성한 사후 예측 분포

$$P(y \mid x, D, \phi)$$

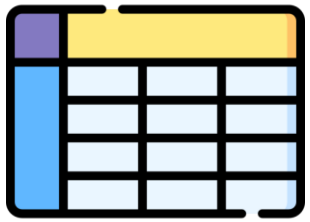
$$\mathcal{L}_{PFN} = \mathop{\mathbb{E}}_{(\{(x_{test}, y_{test})\} \cup D_{train}) \sim p(D)} [-\log q_\theta(y_{test} | x_{test}, D_{train})].$$

$$\hat{P}(y \mid x, D)$$

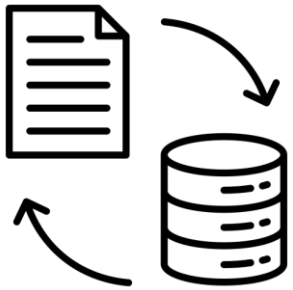
Transformer가 출력한 예측분포

Background

Real-World Inference



$\{x_1, y_1\}$
$\{x_2, y_2\}$
$\{x_3, y_3\}$
...
$\{x_{test},\ \}$



학습된 Transformer

(Parameter Θ 업데이트 X)

$$q_\theta(y \mid x_{\text{test}}, D_{\text{train}})$$

가장 그럴듯한 (Plausible) 사후 예측 분포
(Posterior Predictive Distribution, PPD)

➔ 입력된 데이터만으로 모델이 문맥을 파악하고 예측?
In Context Learning

기존 연구와 차이점

항목	기존 PFN (2022)	TabPFN (2023)
Number of class	Binary	2 ~ 10 개 (Multi class)
Class distribution	Balanced	Imbalanced도 가능
Output	스칼라 → Binary	스칼라 → Multi class 구간화
Prior	BNN based	BNN + SCM 기반, 클래스 샘플링 구조 포함

기존 PFN : 30개의 학습 샘플, 2개의 Class, Test 시 Balanced class 비율을 가진 작은 데이터 셋 대상

TabPFN

1) 정수 분포 $p(N_C)$ 로부터 Class 수를 샘플링

2) 연속적 타겟값 $\hat{y}$에서 N_C-1 의 경계 샘플링하여 기준으로 Class 매핑

3) 무작위로 Class 셔플링 진행

→ Class 수 샘플링 과정에서 Multi class 구현

경계 샘플링, 및 Class 셔플링 과정에서 Imbalance, 분포 편향 영향 제거

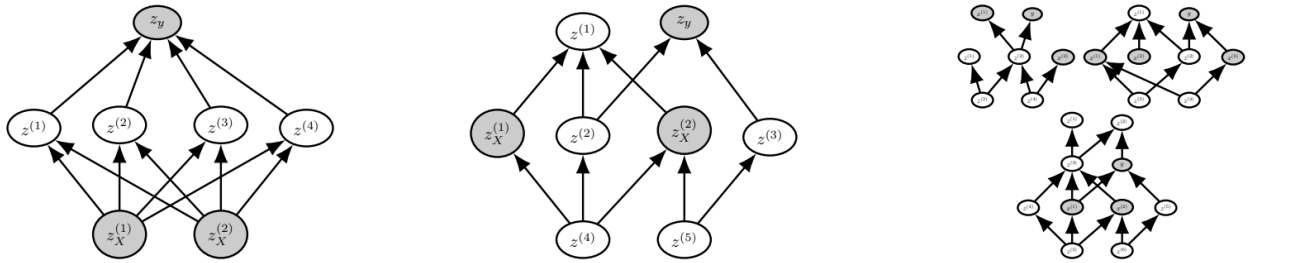
$(-\infty, -0.1]$	Class 1
$(-0.1, 0.5]$	Class 2
$(0.5, \infty)$	Class 3

기존 연구와 차이점

항목	기존 PFN (2022)	TabPFN (2023)
Number of class	Binary	2 ~ 10 개 (Multi class)
Class distribution	Balanced	Imbalanced도 가능
Output	스칼라 → Binary	스칼라 → Multi class 구간화
Prior	BNN based	BNN + SCM 기반

기존 PFN : 샘플링으로 구성된 입력과 출력 사이에 중간 Latent 노드들이 존재하는 일반적인 BNN

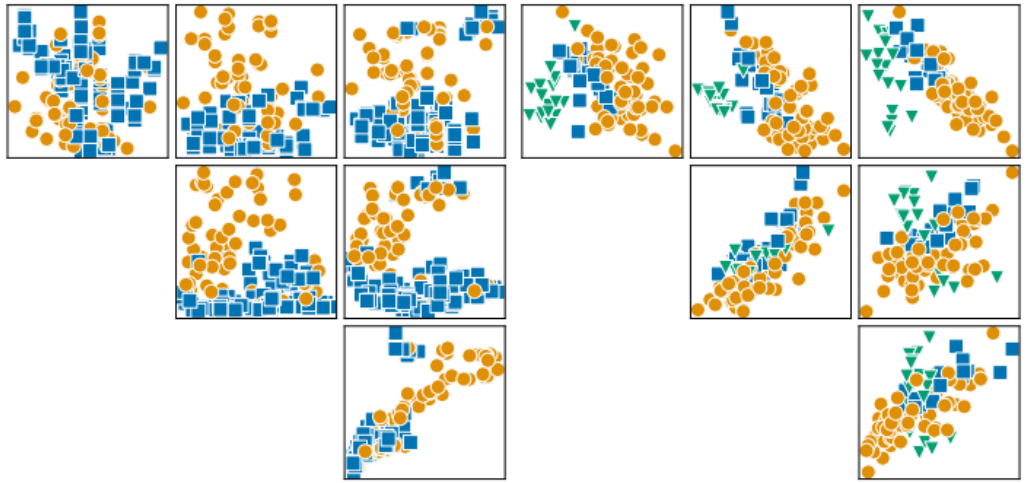
TabPFN



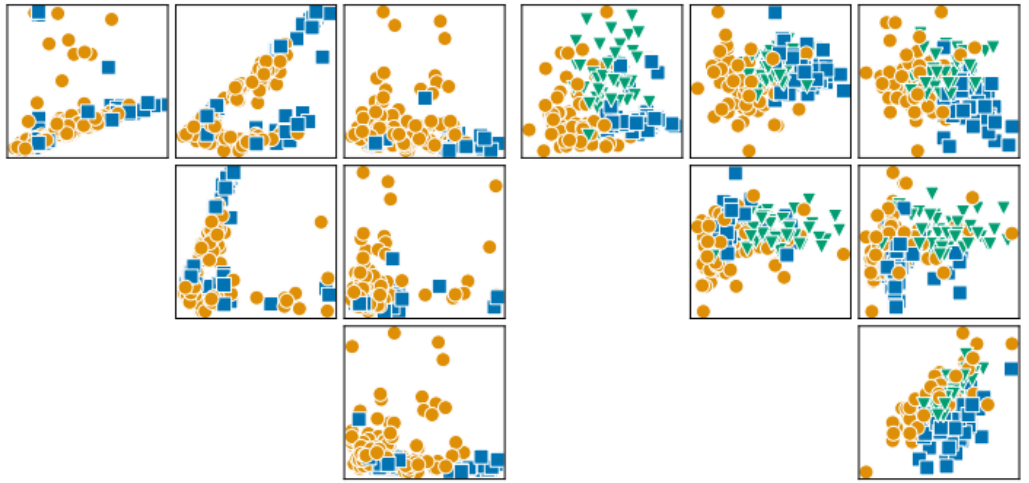
(a) A BNN (b) An SCM (c) SCMs sampled from the prior

Figure 2: Overview of graphs generating data in our prior. Inputs x are mapped to the output y through unobserved nodes z. Plots based on Müller et al. (2022).

SCM Prior



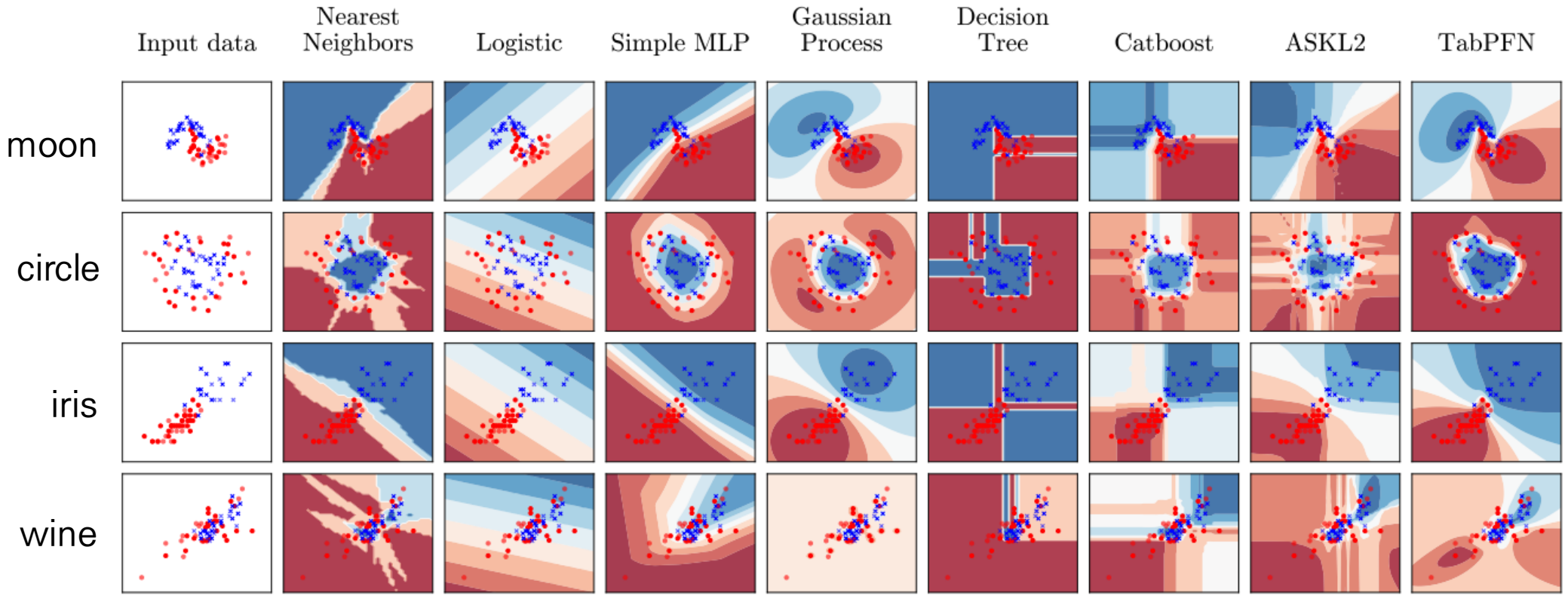
(a) Synthetic datasets



(b) Actual datasets

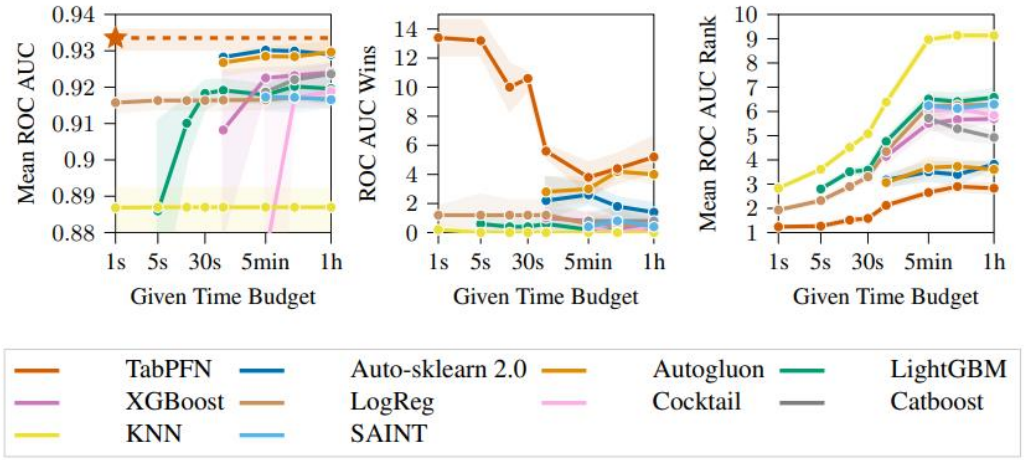
SCM Prior가 생성한 데이터가 실제 데이터처럼 구성됨 (클래스간 분리 가능성 / 구조적 다양성)
→ SCM Prior가 BNN 만 활용할 때보다 일반화 가능성을 높여줌

Decision boundaries on toy datasets



Well-calibrated!

Experiments



	LightGBM	CatBoost	XGBoost	ASKL2.0	AutoGluon	TabPFN$_{n.e.}$	TabPFN	TabPFN + AutoGluon
M. rank AUC OVO	6.9722	4.9444	6.1944	4.4722	4	3.8056	2.9444	**2.6667**
Mean rank Acc.	6.8889	4.9722	6.0556	5.1667	3.8889	3.8889	2.8889	**2.25**
Mean rank CE	5.7778	5.4444	6	6.4167	3.1111	4.1389	3.0278	**2.0833**
Mean AUC OVO	0.92±.013	0.924±.011	0.924±.01	0.929±.0096	0.93±.0091	0.932±.0088	**0.934**±.0086	**0.934**±.0084
Mean Acc.	0.862±.012	0.864±.011	0.866±.011	0.87±.014	0.881±.01	0.873±.0095	0.879±.0089	**0.886**±.0094
Mean CE	0.75±.039	0.747±.029	0.759±.04	0.813±.073	0.714±.014	0.727±.021	0.716±.019	**0.711**±.014
Mean time (s) (Tune + Train + Predict)	3280	3746	3364	3601	3077	**1.301** (CPU) **0.0519** (GPU)	37.59 (CPU) 0.6172 (GPU)	3109 (CPU) 3077 (GPU)

TabPFN은 학습시간이 별도로 존재하지 않음
→ Tune + Train + Predict 측면에서 압도적으로 유리

Experiments

❖ 18개의 결측치 없는 수치형 데이터셋에서의 실험 결과

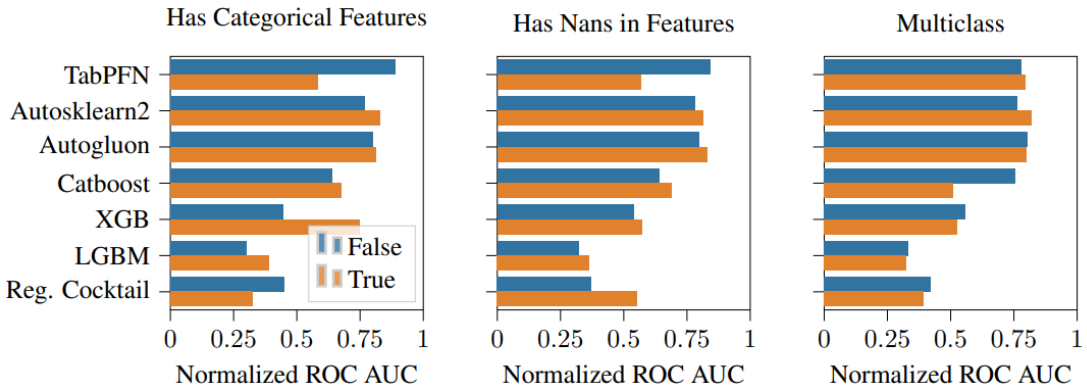
TabPFN은 GPU 상에서 1초 이내의 추론 시간만으로 AutoML 시스템이 1시간 학습한 것과 비슷한 성능을 달성.

튜닝된 GBDT(XGBoost, CatBoost, LightGBM)보다도 성능 우수.

❖ TabPFN은 비슷한 성능의 모델 대비 훨씬 빠름

CPU 기준 230배, GPU 기준 5,700배 빠름.

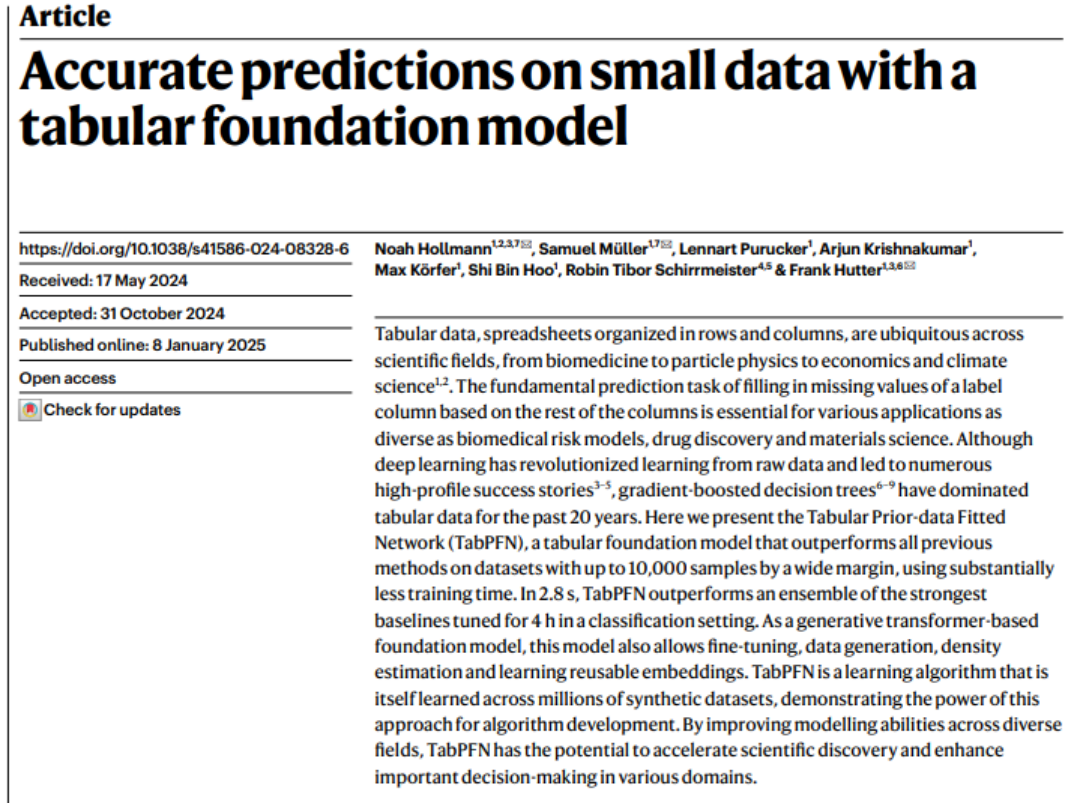
단, 범주형 변수나 결측값이 포함된 경우 성능이 낮을 수 있음.



Transformer-Based Bayesian Inference for Tabular Data

❖ **Accurate predictions on small data with a tabular foundation model**

- Nature 2025 게재, 25년 7월 기준 138회 인용



Article

Accurate predictions on small data with a tabular foundation model

https://doi.org/10.1038/s41586-024-08328-6

Received: 17 May 2024

Accepted: 31 October 2024

Published online: 8 January 2025

Open access

Check for updates

Noah Hollmann[1,2,3,7], Samuel Müller[1,7], Lennart Purucker[1], Arjun Krishnakumar[1], Max Körfer[1], Shi Bin Hoo[1], Robin Tibor Schirrmeister[4,5] & Frank Hutter[1,3,6]

Tabular data, spreadsheets organized in rows and columns, are ubiquitous across scientific fields, from biomedicine to particle physics to economics and climate science[1,2]. The fundamental prediction task of filling in missing values of a label column based on the rest of the columns is essential for various applications as diverse as biomedical risk models, drug discovery and materials science. Although deep learning has revolutionized learning from raw data and led to numerous high-profile success stories[3–5], gradient-boosted decision trees[6–9] have dominated tabular data for the past 20 years. Here we present the Tabular Prior-data Fitted Network (TabPFN), a tabular foundation model that outperforms all previous methods on datasets with up to 10,000 samples by a wide margin, using substantially less training time. In 2.8 s, TabPFN outperforms an ensemble of the strongest baselines tuned for 4 h in a classification setting. As a generative transformer-based foundation model, this model also allows fine-tuning, data generation, density estimation and learning reusable embeddings. TabPFN is a learning algorithm that is itself learned across millions of synthetic datasets, demonstrating the power of this approach for algorithm development. By improving modelling abilities across diverse fields, TabPFN has the potential to accelerate scientific discovery and enhance important decision-making in various domains.

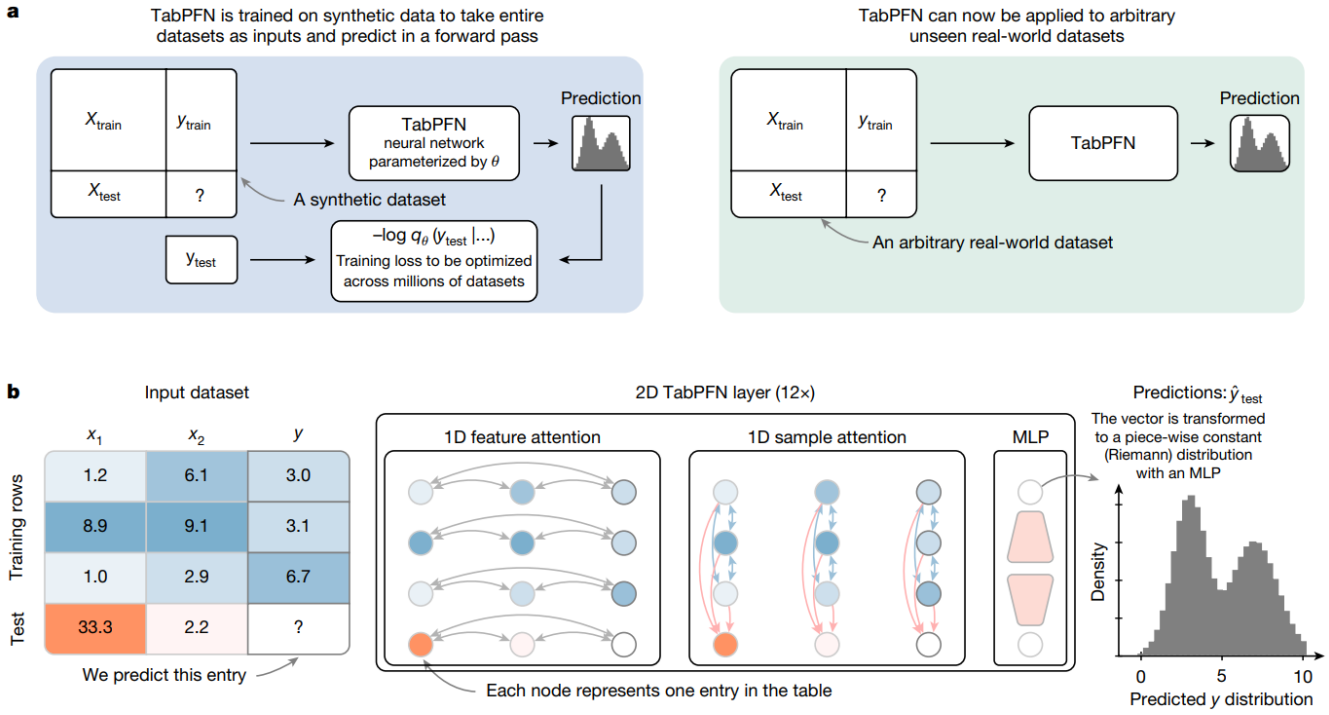
Background

❖ **TabPFN 논문으로 Foundation model for tabular data의 가능성 확인**

한계

1. 입력토큰을 시퀀스로 간주함
   - Tabular의 구조 (row/column) 반영 X

2. Fit-predict 재사용 불가
   - Train/testset을 동시에 입력하는 구조라 testset 변경시 매 번 다시 계산해야함



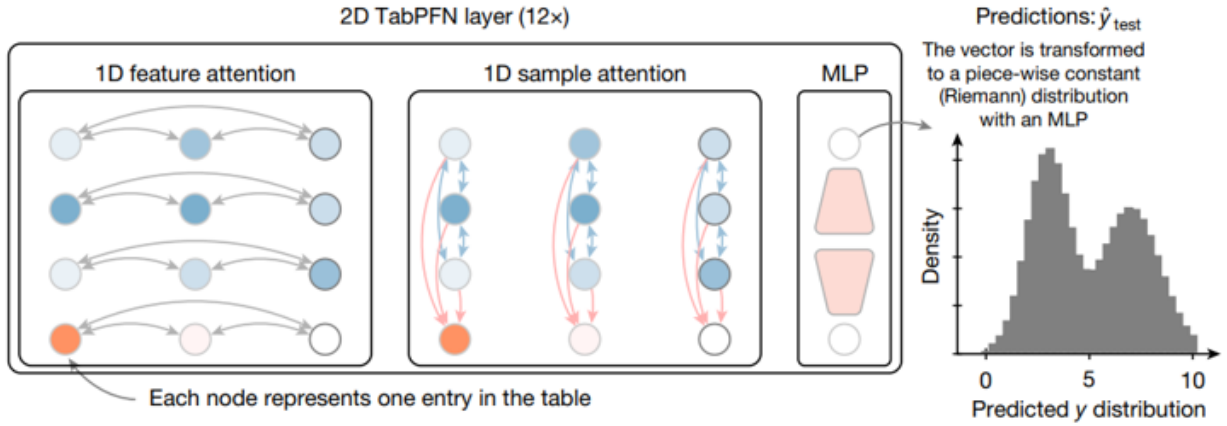
TabPFN V2

❖ **TabPFN 논문으로 Foundation model for tabular data의 가능성 확인**

한계

1. 입력토큰을 시퀀스로 간주함
    - Tabular의 구조 (row/column) 반영 X

✓ 각 셀에 독립적 표현 부여 및 2-way attention 같은 row/colomn내 다른 셀과 상호작용하여 순서에 대한 불변성 유지 및 확장 가능



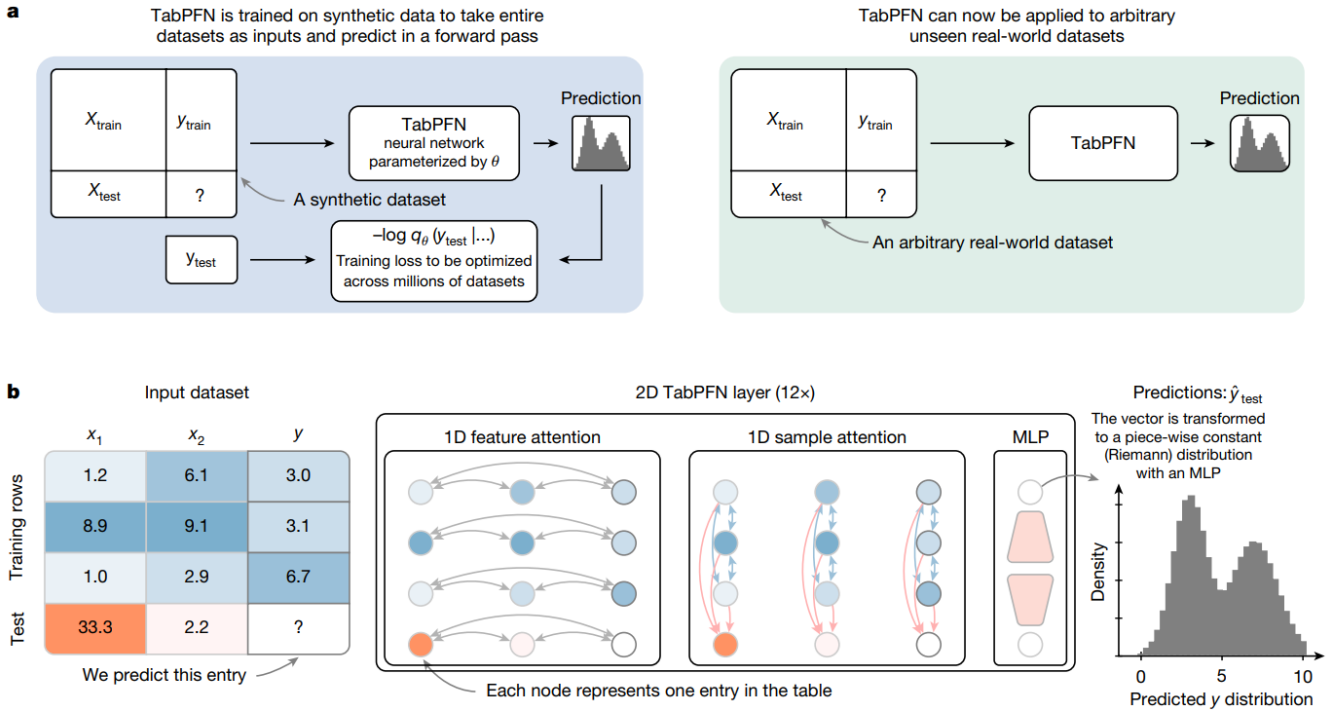
TabPFN V2

❖ **TabPFN 논문으로 Foundation model for tabular data의 가능성 확인**

한계

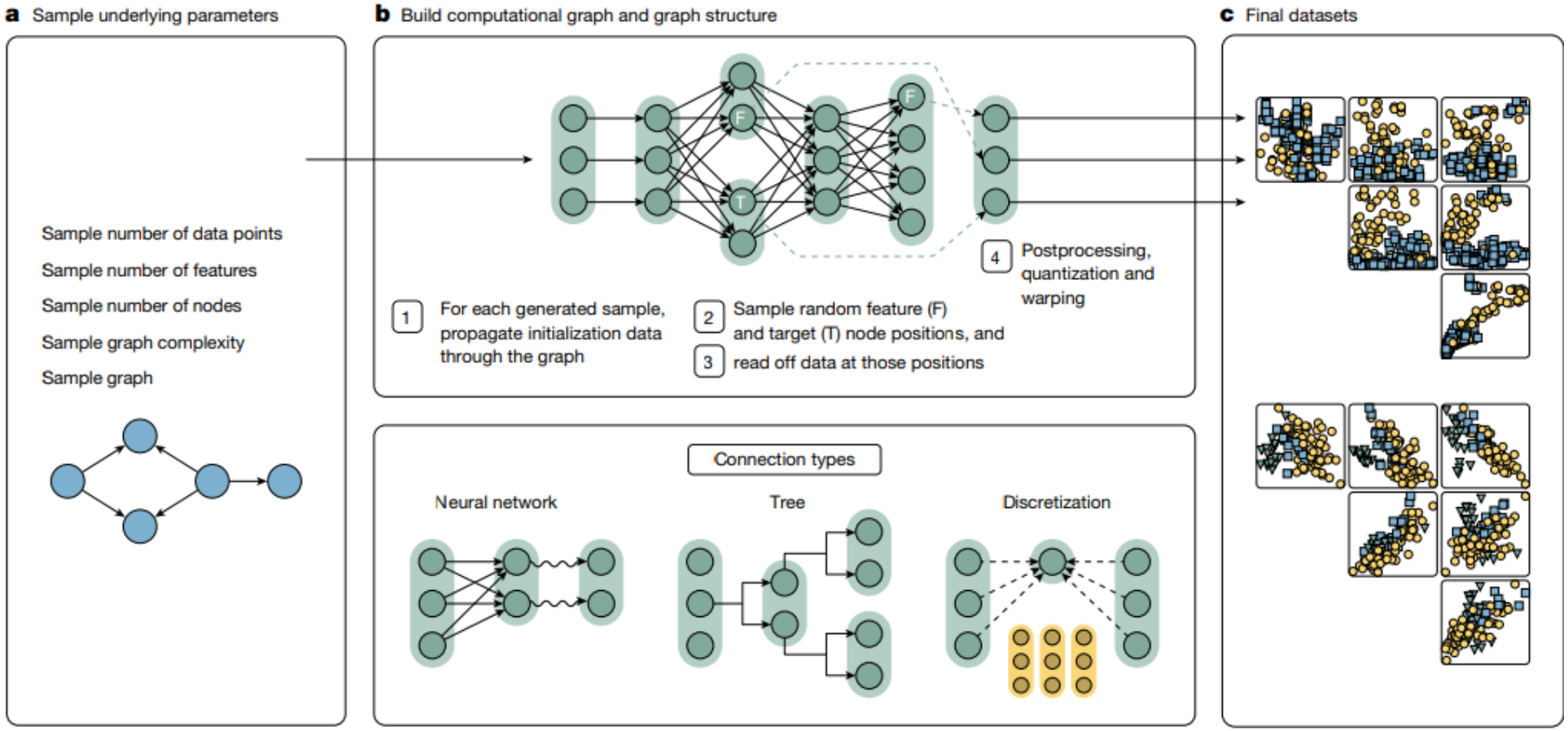
2. Fit-predict 재사용 불가
   - Train/testset을 동시에 입력하는 구조라 testset 변경시 매 번 다시 계산해야함

✓ Trainset의 Attention 계산 결과를 Cache 저장
→ Testset이 변경되어도 빠르게 추론



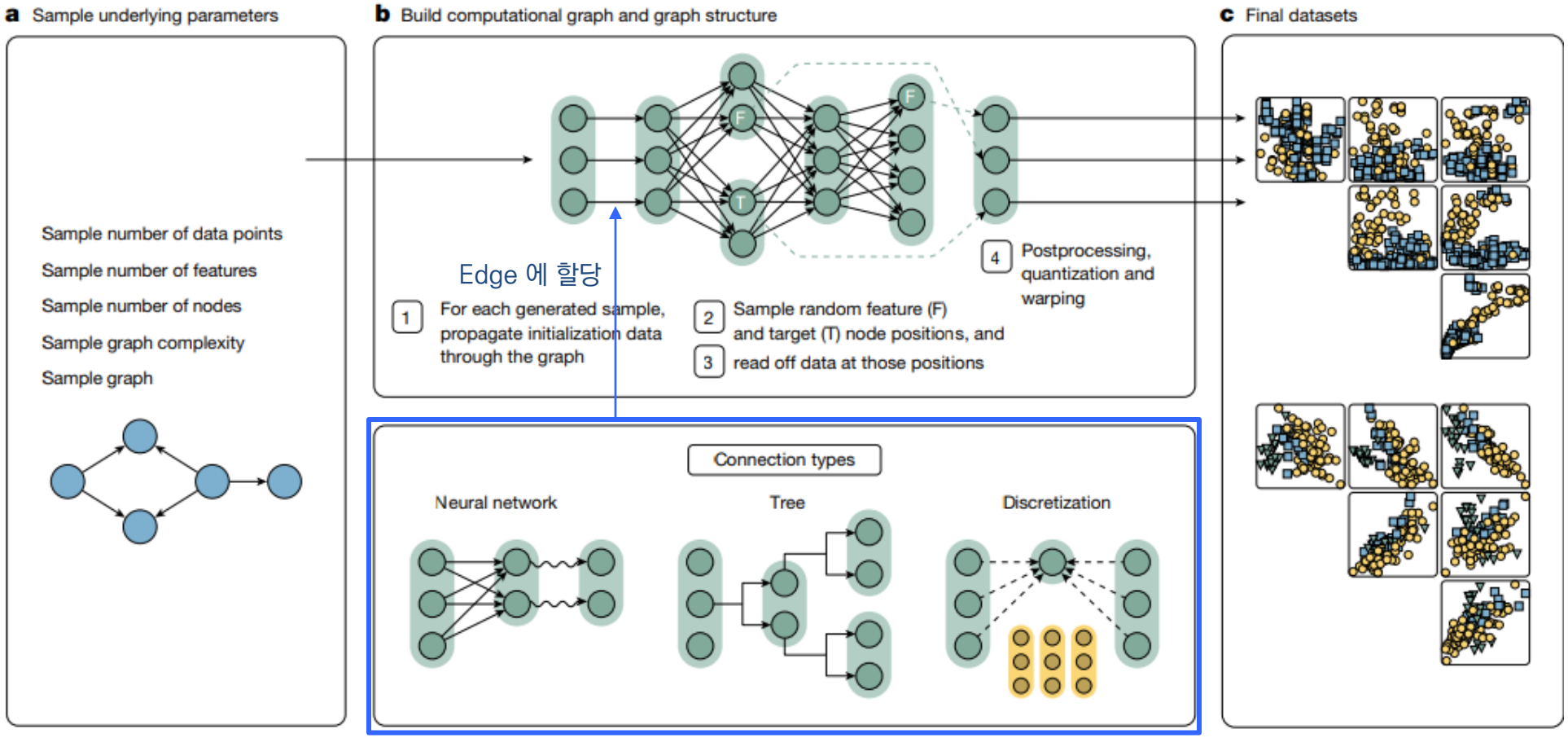
TabPFN V2

Prior 생성



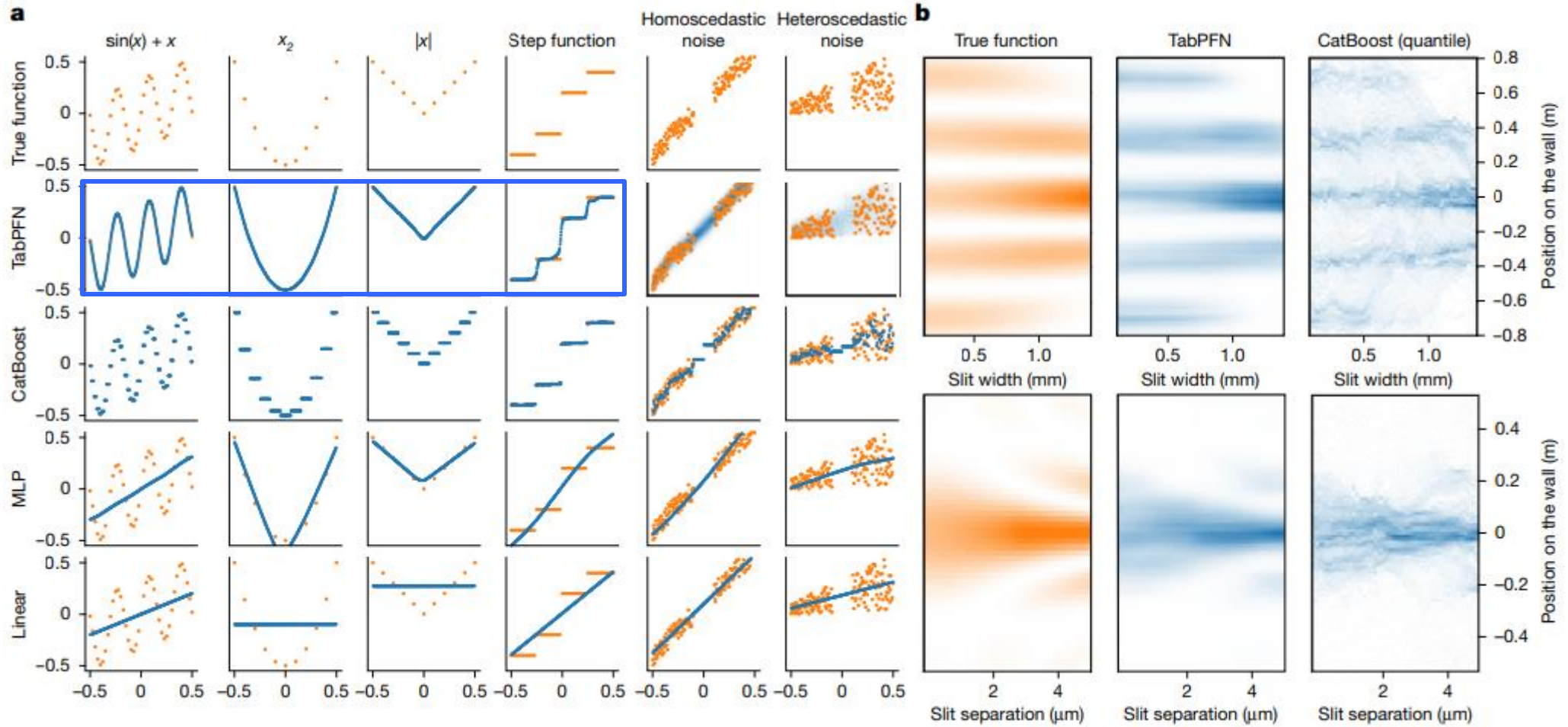
TabPFN V2

Prior 생성



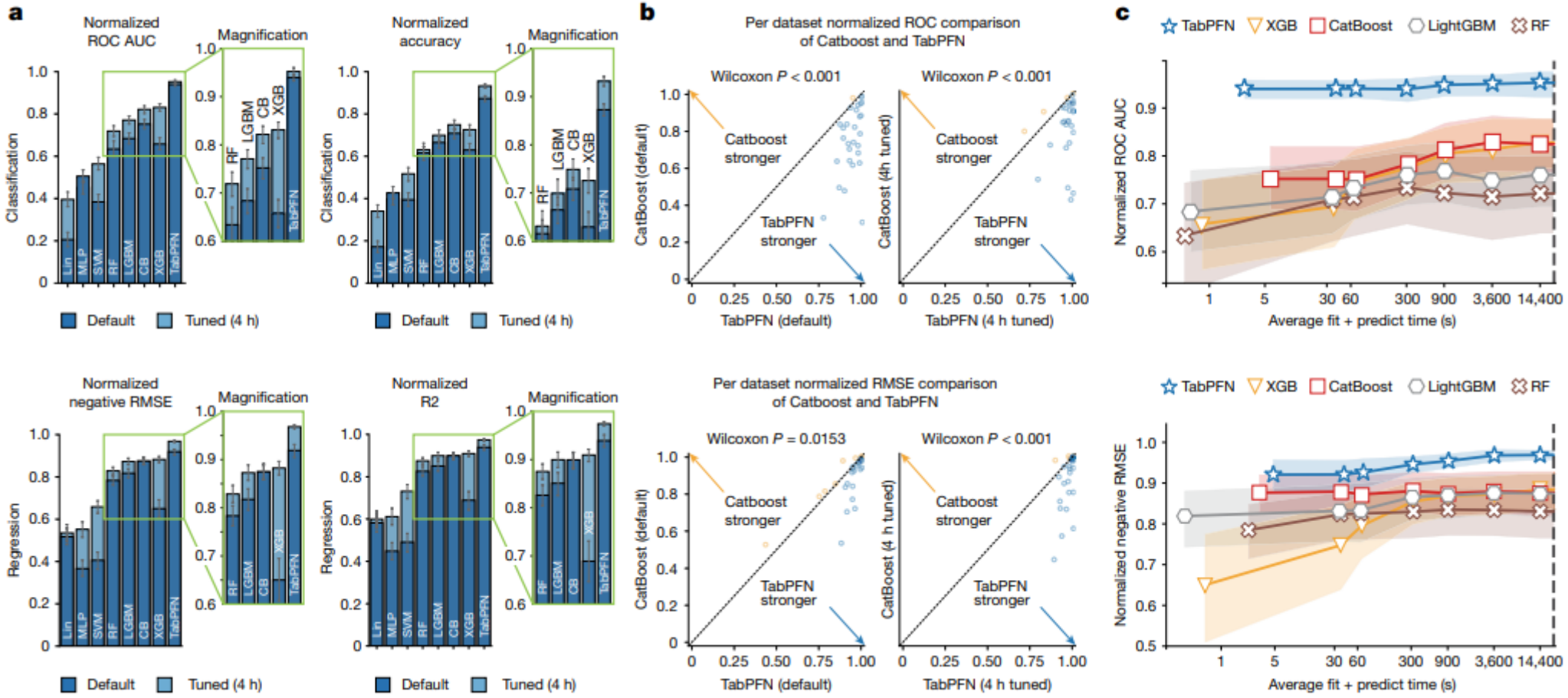
Experiments

Qualitative analysis



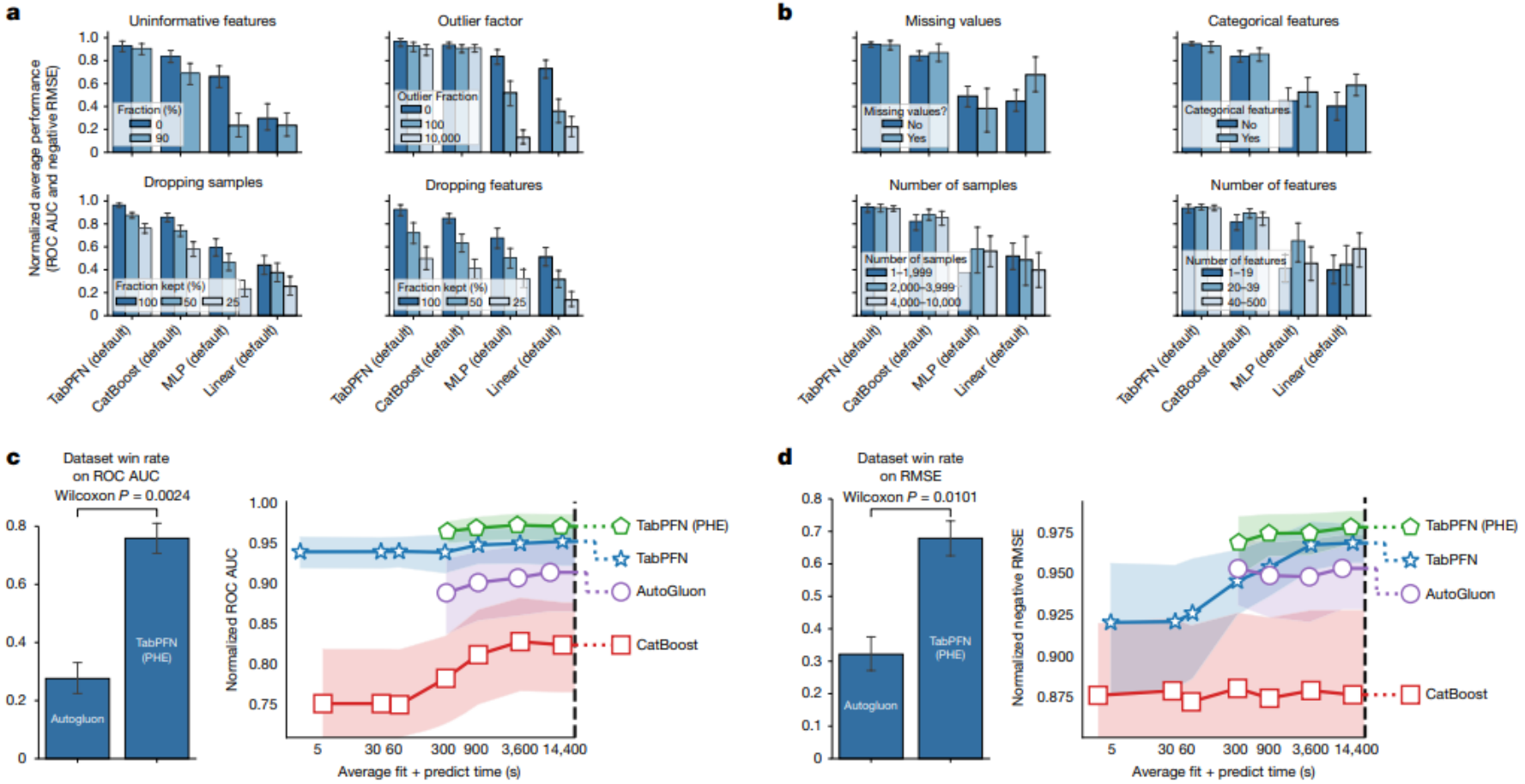
Experiments

Quantitative analysis



Experiments

Robustness 및 Ensemble 성능 확장



Conclusion

❖ **Transformers Can Do Bayesian Inference**

- Transformer가 사전 분포와 데이터를 결합해 사후 예측 분포(PPD)를 근사하여, 베이지안 추론 문제를 In-Context Learning으로 해결함을 보여 PFN(Prior-Data Fitted Network)의 이론적 기반을 마련

❖ **TabPFN: A Transformer that solves small tabular classification problems in a second**

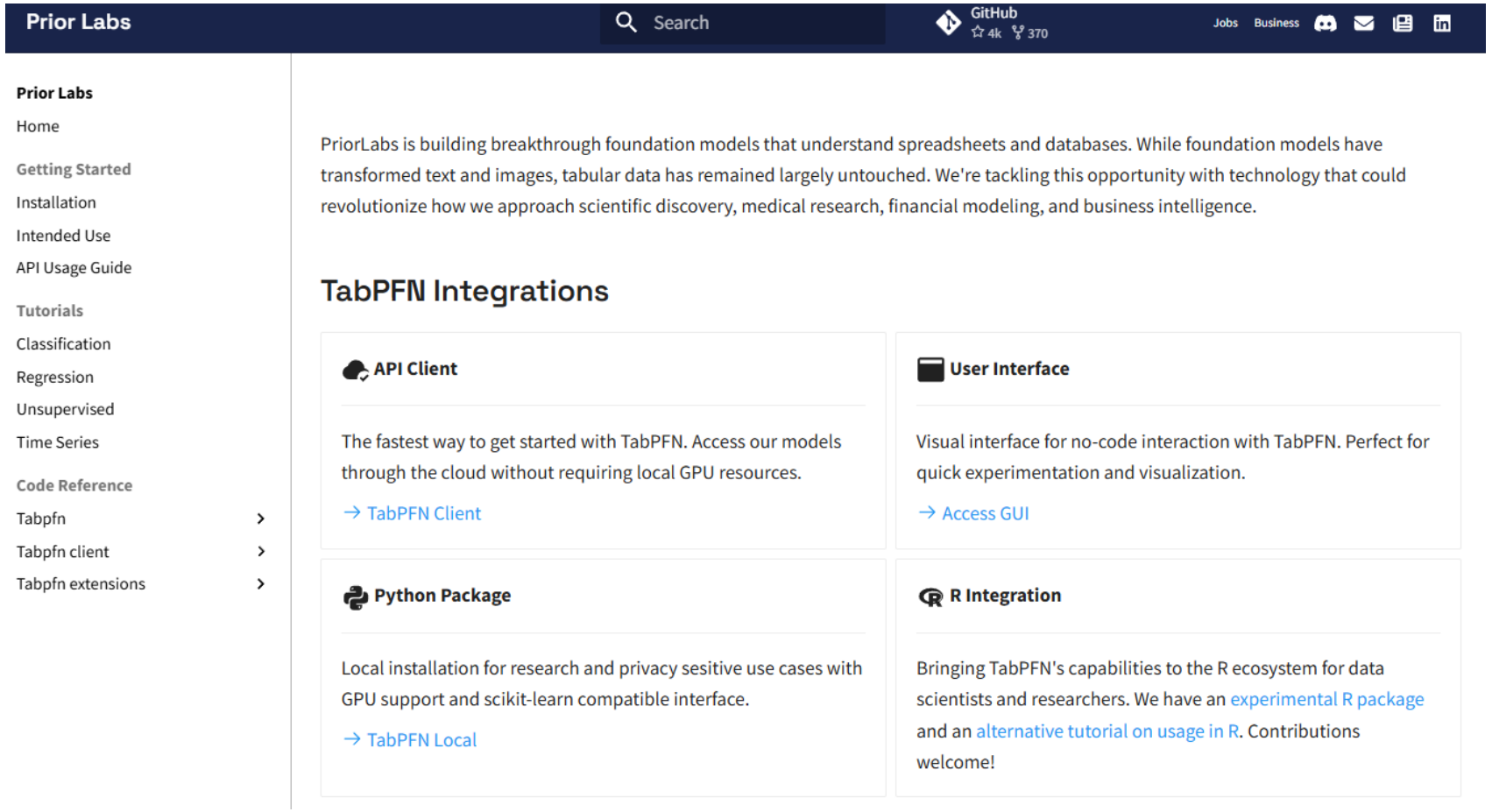
- TabPFN은 소규모 tabular 분류 문제를 위해 사전 학습된 Transformer를 사용해, 단일 forward pass만으로 빠르게 예측이 가능한 ICL 기반의 새로운 패러다임을 제시

❖ **Accurate predictions on small data with a tabular foundation model**

- SCM 기반의 대규모 synthetic prior를 학습하여, tabular foundation model로서 기존 SOTA 모델을 웃도는 성능과 효율성을 입증

Appendix

Prior lab



Thank you