TabNet

Tags	
Property	@February 22, 2021 3:36 PM
references	
열	

1. Introduction

- 심층 신경망 (DNN)은 이미지 (He et al. 2015), 텍스트 (Lai et al. 2015) 및 오디오 (Amodei et al. 2015)에서 주목할만한 성공을 보여줌
- 그러나 테이블 형식 데이터는 실제 가장 일반적인 데이터 유형임에도 불구하고 아직 주목할만한 성공을 보여주지 못함
- 앙상블 의사 결정 트리(DTs)의 변형을 사용하여 테이블 형식 데이터에 대한 딥러닝은 아직 탐구되지 않았지만 Kaggle 을 보면 대부분의 애플리케이션을 지배하고 있음

▼ 왜?

1) DT 기반 접근 방식의 장점 때문.

DT 기반 접근 방식의 장점

- 테이블 형식 데이터에서 일반적으로 사용되는 대략적인 초평면 경계가 있는 의사 결정 manifolds에 대해 효율적
- 기본 형식 (예 : 의사 결정 노드 추적)에서 매우 해석 가능하며 앙상블 형식과 같은 사후 설명 방법 존재
- 훈련하기가 빠릅니다

2) 이전에 제안된 DNN 아키텍처는 테이블 형식 데이터에 적합하지 않기 때문

여러 개로 쌓인 convolution layer 또는 다중 레이어 퍼셉트론 (MLP)은 매개 변수가 매우 많아지므로 적절한 inductive bias의 부족으로 인해 테이블 형식 decision manifolds에 대한 최적값을 찾지 못하는 경우가 많음

▼ 딥러닝에서 표형식 데이터tabular data를 탐색할 가치가있는지?

대규모 데이터 세트의 예상 성능 향상을 위해 필요. 트리학습과 달리 DNN은 다양한 이점을 가짐.

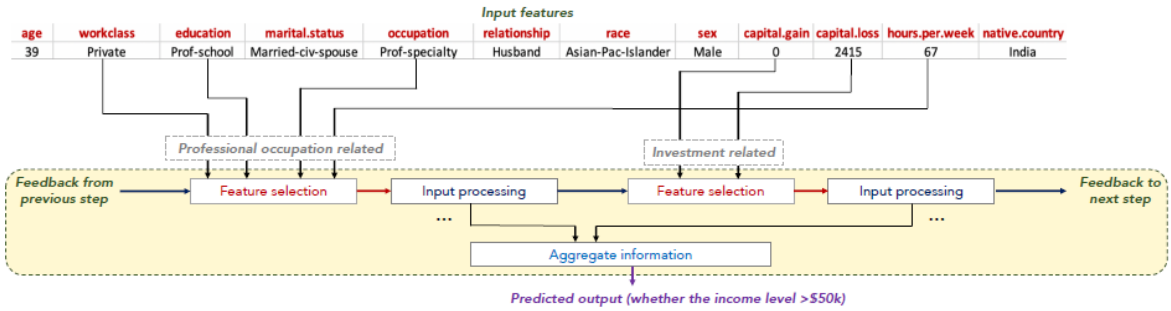
- 표 형식 데이터와 함께 이미지와 같은 여러 데이터 유형을 효율적으로 인코딩 가능.
- 현재 트리 기반 테이블 형식 데이터 학습 방법의 핵심 요소인 기능 엔지니어링의 필요성 완화
- 스트리밍 데이터의 Learning
- 테이블 형식 데이터에 대한 경사 하강법 기반의 종단 간 학습을 가능하게 함.
- end-to-end 모델은 데이터 효율적인 도메인 적응을 포함하여 많은 가치있는 애플리케이션 시나리오를 가능하게 하는 표현 학습을 허용합니다

논문의 기여점

테이블 형식 데이터를 처리하는 TabNet을 위한 새로운 표준 DNN 아키텍처를 제안

1. TabNet은 사전 처리없이 원시 테이블 형식 데이터(raw table data)를 입력하고 경사 하강법 기반 최적화를 사용하여 학습되므로 종단 간 학습(end-to-end)에 유연하게 통합 할 수 있습니다.

2. TabNet은 sequential attention를 사용하여 각 의사 결정 단계에서 추론할 function을 선택하여 학습 능력이 가장 두드러진 기능에 사용되므로 해석 가능성과 더 나은 학습을 가능하게 합니다 (그림 1 참조).



(그림1. 성인 인구 조사 소득 예측 (Dua and Graff 2017)에 예시 된 TabNet의 Sparse feature selection.

1) Sparse feature selection은 용량이 가장 두드러진 function에 사용되므로 해석 가능성과 더 나은 학습을 가능하게 함.

2) TabNet은 추론을 위해 입력 function의 하위 집합을 처리하는 데 초점을 맞춘 여러 의사 결정 블록을 사용.

- 예시로 표시된 두 가지 결정 블록은 소득 수준을 예측하기 위해 각각 전문 직업 및 투자와 관련된 기능을 처리
- 이 feature selection은 인스턴스별로 instance-wise로 적용. e.g. 입력마다 다를 수 있으며 (Chen et al. 2018) 또는 (Yoon, Jordon 및 van der Schaar 2019)와 같은 다른

인스턴스 별 function 선택 방법과 달리 TabNet은 function 선택 및 **추론**을 위해 단일 딥러닝 아키텍처를 사용.

3) 위의 feature selection 은 두 가지 중요한 속성으로 이어짐.

1- TabNet은 다른 도메인의 분류 및 회귀 문제에 대해 다양한 데이터 세트에서 다른 테이블 형식 학습 모델보다 성능이 우수하거나 동등합니다.

2- TabNet은 두 가지 종류의 해석성을 지원

- local 해석 가능성 : function의 중요성과 결합 방법을 시각화
- global 해석 가능성 : 훈련된 모델에 대한 각 기능의 기여도를 정량화

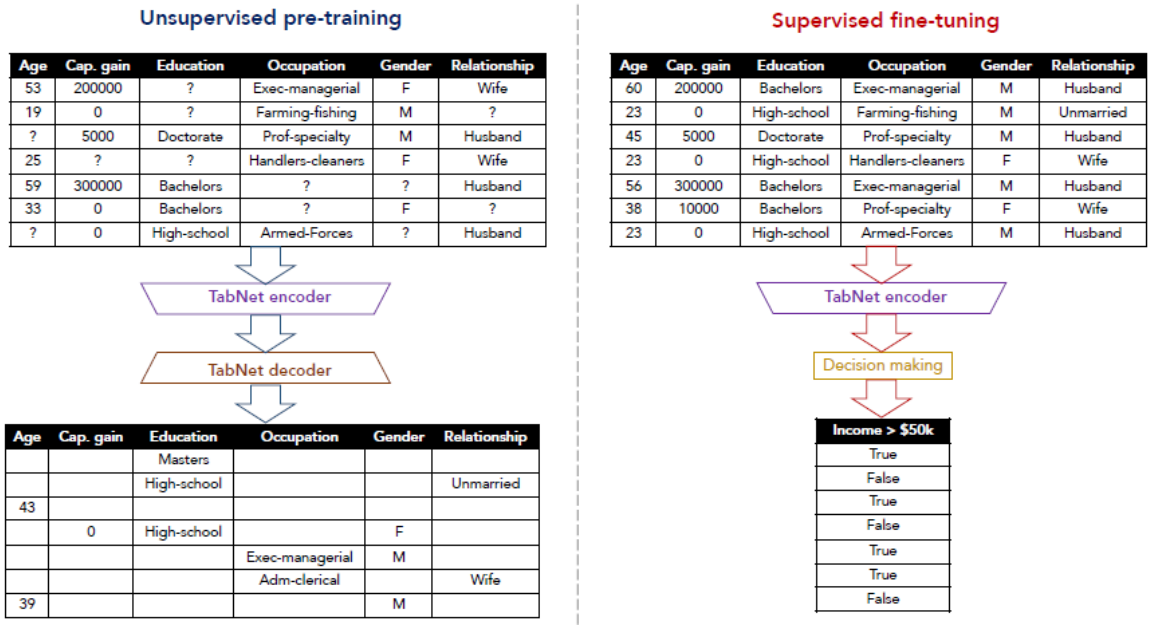
4) 표 형식 데이터의 경우 감독되지 않은 사전 훈련(unsupervised-pretraining)을 사용하여 마스킹된 function을 예측함으로써 상당한 성능 향상을 보여줍니다 (그림 2 참조).

▼ function : table 의 각 원소

(그림2 : Self-supervised tabular learning. 실제 테이블 형식 데이터 세트에는 상호 의존적인 특성 열이 있음.

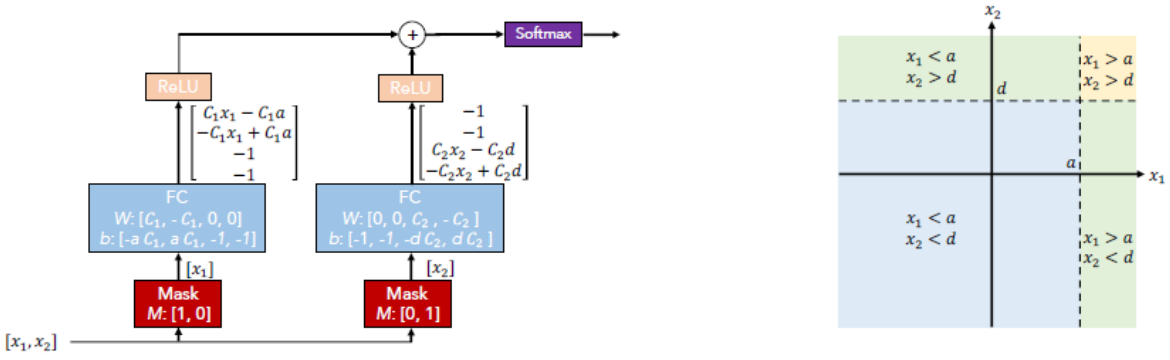
- 예를 들어 직업에서 교육 수준을 추측하거나 관계에서 성별을 추측 할 수 있습니다.

마스킹된 self-supervised learning에 의한 unsupervised representation 학습은 supervised learning task을 위한 인코더 모델을 개선



2. TabNet for Tabular Learning

DT(decision tree)는 실제 테이블 형식 데이터 세트에서 학습하는 데 성공적입니다. 특정 디자인의 경우 기존 DNN building 블록을 사용하여 DT와 유사한 출력 매니 폴드를 구현할 수 있습니다. (그림 3 참조).



(그림3 : 기존 DNN 블록 (왼쪽)과 해당 결정 매니 폴드 (오른쪽)를 사용한 DT 유사 분류의 그림. 입력에 multiplicative sparse masks를 사용하여 관련 features을 선택.

선택된 특징은 선형으로 변환되고, 바이어스 추가 (경계를 나타 내기 위해) 후에 ReLU는 영역을 0으로 설정하여 영역 선택을 수행합니다.

여러 지역의 집계는 addition을 기반으로 합니다. (C1과 C2가 커질수록 결정 경계가 더 선명 해집니다.)

이러한 설계에서 개별 특징 선택individual feature selection은 하이퍼 플레인 형태의 결정 경계를 얻기위한 핵심이며, 계수가 각 특징의 비율을 결정하는 특징의 선형 조합으로 일반화 될 수 있습니다.

TabNet은 이러한 기능을 기반으로하며 신중한 설계로 이점을 누리면서 DT를 능가합니다.

1) 데이터에서 학습한 희소 인스턴스 별 features selection을 사용합니다.

2) 순차적 다단계 아키텍처를 구성합니다. 여기서 각 단계는 선택한 기능에 따라 결정의 일부에 기여합니다.

3) 선택한 기능의 비선형 처리를 통해 학습 능력 향상

4) 더 높은 차원과 더 많은 단계를 통해 앙상블을 모방합니다.

그림 4는 표 형식 데이터를 인코딩하기위한 TabNet 아키텍처를 보여줍니다.

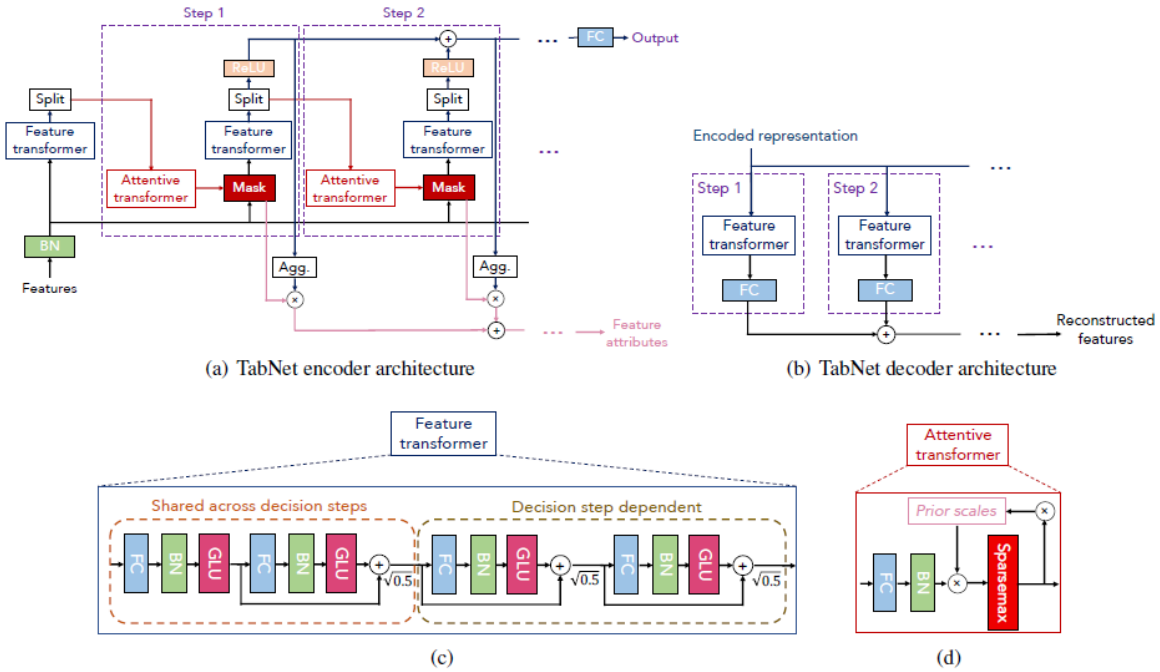


그림4: (a) feature transformer, attentive transformer, feature masking으로 구성된 TabNet 인코더. split block은 후속 단계의 attentive transformer와 전체 출력에 사용할 처리된 표현을 나눕니다. 각 단계에서 feature selection masking는 모델의 functionality에 대한 해석 가능한 정보를 제공하며 마스크를 집계하여 global feature important attribution을 얻을 수 있습니다.

(b) 각 단계에서 feature transformer 블록으로 구성된 TabNet 디코더.

(c) feature transformer 예 – 4 layer 네트워크가 표시됩니다. 여기서 2 개는 all decision steps에서 공유되고 2 개는 decision steps에 따라 다릅니다. 각 계층은 완전 연결 (FC) 계층, BN 및 GLU 비선형성으로 구성.

(d) attentive transformer block 예-단일 레이어 매핑은 현재 결정 단계 전에 각 기능이 얼마나 많이 사용되었는지 집계하는 이전 스케일 정보로 변조됩니다. sparsemax(Martins and Astudillo 2016)는 계수의 정규화에 사용되어 두드러진 특징을 희소하게 선택합니다.

raw numerical features을 사용하고 학습 가능한 임베딩을 사용한 범주 특성 매핑을 고려합니다. 어떤 전역 기능 정규화global feature normalization도 고려하지 않고 배치 정규화 (BN) 만 적용합니다. 동일한 D 차원 features를 각 결정decision 단계에 전달합니다. 여기서 B는 배치 크기입니다.

TabNet의 인코딩은 N_steps 결정 단계가있는 순차적인 다단계 처리를 기반으로합니다.

i 번째 단계는 (i-1) 번째 단계에서 처리된 정보를 입력하여 사용할 features을 결정하고 processed features 표현을 전체 결정decision으로 집계할 출력을 합니다. 순차sequential 형식의 top-down attention 개념은 고차원 입력에서 관련 정보의 작은 하위 집합을 검색하

는 동안 시각적 및 텍스트 데이터 처리(Hudson and Manning 2018) 및 강화 학습(Mott et al. 2019)에 적용한 응용 프로그램에서 영감을 받았습니다.

Feature selection

두드러진 특징을 **부드럽게 선택**하기 위해 학습 가능한 마스크를 사용.

$$M[i] \in R^{B \times D}$$

가장 두드러진 특징의 **희박한 선택**을 통해 의사 결정 단계의 학습 능력이 관련없는 것에 낭비되지 않으므로 모델이 더 효율적으로 매개 변수가됩니다. 마스킹은 곱셈으로 이뤄짐

$$M[i] \cdot f$$

attentive transformer (그림 4 참조)를 사용하여 마스크를 얻습니다.

이전 단계에서 처리 된 기능

$$a[i-1]; M[i] = sparsemax(P[i-1] \cdot h_i(a[i-1]))$$

Sparsemax 정규화(https://towardsdatascience.com/what-is-sparsemax-f84c136624e4)는 유클리드 투영을 확률적 심플렉스에 매핑하여 희소성을 생성합니다.(ref. https://towardsdatascience.com/what-is-sparsemax-f84c136624e4)

Algorithm 1 Sparsemax Evaluation

Input: z

Sort z as $z_{(1)} \geq \ldots \geq z_{(K)}$

Find $k(z) := \max \left\{ k \in [K] \mid 1 + kz_{(k)} > \sum_{j \leq k} z_{(j)} \right\}$

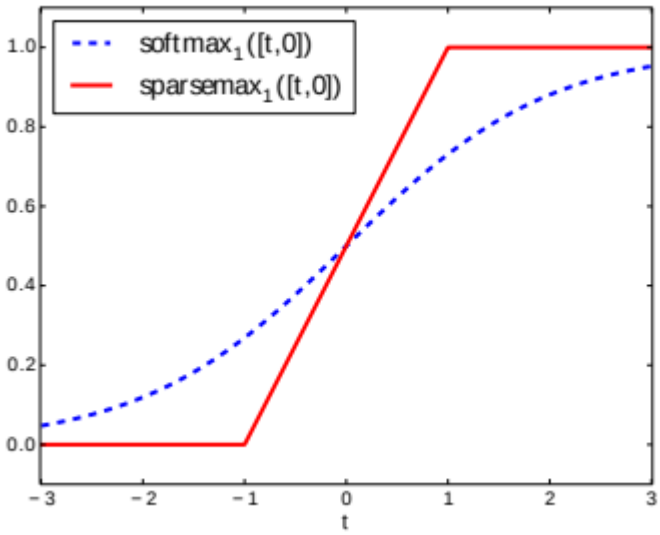
Define $\tau(z) = \frac{\left(\sum_{j \leq k(z)} z_{(j)}\right) - 1}{k(z)}$

Output: p s.t. $p_i = [z_i - \tau(z)]_+$.

```
def sparsemax(z):
    sum_all_z = sum(z)
    z_sorted = sorted(z, reverse=True)
    k = np.arange(len(z))
    k_array = 1 + k * z_sorted
    z_cumsum = np.cumsum(z_sorted) - z_sorted
```

```
    k_selected = k_array > z_cumsum
    k_max = np.where(k_selected)[0].max() + 1
    threshold = (z_cumsum[k_max-1] - 1) / k_max
    return np.maximum(z-threshold, 0)
```

이는 성능이 우수하고 설명 가능성을 위한 희소 특징 선택의 목표와 일치.



$$\sum_{j=1}^{D} M[i]_{b,j} = 1$$

에서 h_i 는 훈련 가능한 함수이며 FC 계층을 사용하는 그림 4와 BN이 뒤 따름. P[i]는 특정 기능이 이전에 얼마나 많이 사용되었는지를 나타내는 이전 척도 용어.

$$P[i] = \prod_{j=1}^{i} (\gamma - M[j])$$

여기서 gamma 는 이완 매개 변수이며 gamma=1 일 때, feature은 하나의 의사 결정 단계에서만 사용되도록 강제되며 가 증가함에 따라 여러 의사 결정 단계에서 기능을 사용하기 위해 더 많은 유연성이 제공됩니다.

P[0] 은 any prior on the masked features 없이 모두 1로 초기화됩니다.

$$1^{B \times D}$$

일부 features이 사용되지 않는 경우 (supervised learning 학습에서와 같이) 해당 항목이 0이되어 모델 학습을 돕습니다.

선택된 feature 의 희소성을 **더 제어하기 위해** 우리는 엔트로피 형태의 희소성 정규화를 제안합니다

$$L_{sparse} = \sum_{i=1}^{N_{steps}} \sum_{b=1}^{B} \sum_{j=1}^{D} \frac{-M_{b,j}[i] log(M_{b,j}, [i] + \epsilon)}{N_{steps} \cdot B}$$

여기서 epsilon은 수치 안정성을 위한 작은 숫자입니다. 계수 **lambda_{sparse} = L_{sparse}**를 사용하여 전체 손실에 희소성 정규화를 추가합니다.

희소성은 대부분의 features이 중복되는 데이터 세트에 유리한 유도 편향inductive bias을 제공합니다.

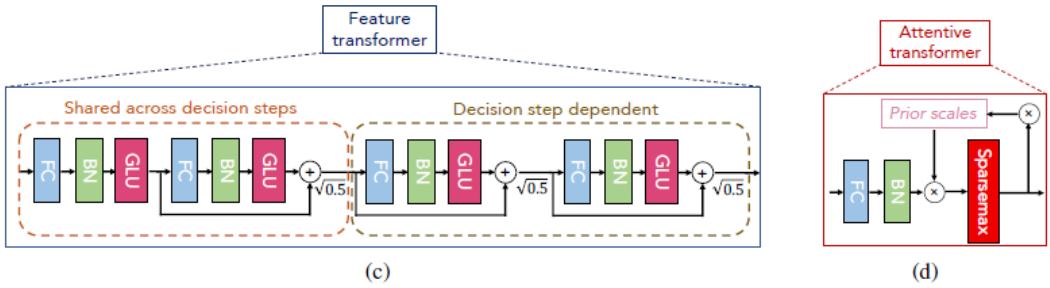
Feature processing

feature transformer (그림 4 참조)를 사용하여 필터링 된 특성을 처리 한 다음 결정 단계 출력 및 후속 단계에 대한 정보를 위해 분할합니다.

$$[d[i], a[i]] = f_i(M[i] \cdot f), where \;\; d[i] \in R^{B \times N_d}$$

고 용량high capacity의 매개 변수를 효율적이고 강력한 학습을 위해 features transformer는 모든 의사 결정 단계에서 공유되는 레이어 (동일한 features이 다른 의사 결정 단계에 입력 됨)와 의사 결정 단계 종속 레이어로 구성되어야합니다.

그림 4는 2개의 공유 계층과 2개의 결정 단계 종속 계층의 연결로 구현을 보여줍니다.



각 FC 계층 뒤에는 BN 및 게이트 선형 단위 (GLU) 비선형성(Dauphin et al. 2016)이 뒤 따르며 결국 정규화를 통해 정규화된 residual 연결에 연결됩니다.

sqrt{0.5} 정규화는 네트워크 전체의 분산이 크게 변경되지 않도록하여 학습을 안정화하는 데 도움이됩니다.

더 빠른 훈련을 위해 BN과 함께 큰 배치 크기를 사용하며, 입력 특성에 적용된 것을 제외하고 가상 배치 크기 및 운동량 를 사용하여 **ghost BN** 형식을 사용

- **ghost BN 의 seudo-code**

Algorithm 1: Ghost Batch Normalization (GBN), applied to activation x over a large batch B_L with virtual mini-batch B_S. Where $B_S < B_L$.

Require: Values of x over a large-batch: $B_L = \{x_{1...m}\}$ size of virtual batch $|B_S|$; Parameters to be learned: γ, β, momentum η

Training Phase:

Scatter B_L to $\{X^1, X^2, ...X^{|B_L|/|B_S|}\} = \{x_{1...|B_S|}, x_{|B_S|+1...2|B_S|} \cdots x_{|B_L|-|B_S|...m}\}$

$\mu_B^l \leftarrow \frac{1}{|B_S|}\sum_{i=1}^{|B_S|} X_i^l$ for $l = 1, 2, 3 \ldots$ {calculate ghost mini-batches means}

$\sigma_B^l \leftarrow \sqrt{\frac{1}{|B_S|}\sum_{i=1}^{|B_S|}(X_i^l - \mu_B)^2 + \epsilon}$ for $l = 1, 2, 3 \ldots$ {calculate ghost mini-batches std}

$\mu_{run} = (1-\eta)^{|B_S|}\mu_{run} + \sum_{i=1}^{|B_L|/|B_S|}(1-\eta)^i \cdot \eta \cdot \mu_B^l$

$\sigma_{run} = (1-\eta)^{|B_S|}\sigma_{run} + \sum_{i=1}^{|B_L|/|B_S|}(1-\eta)^i \cdot \eta \cdot \sigma_B^l$

return $\gamma\frac{X^l - \mu_B^l}{\sigma_B^l} + \beta$

Test Phase:

return $\gamma\frac{X - \mu_{run}^l}{\sigma_{run}} + \beta$ {scale and shift}

input features의 경우 low-variance averaging의 이점을 관찰하므로 **ghost** BN을 피합니다.

마지막으로, 그림 3과 같이 집계와 같은 의사 결정 트리에서 영감을 얻어 전체 의사 결정 임베딩을

$$d_{out} = \sum_{i=1}^{N_{steps}} ReLU(d[i])$$

로 구성합니다. 출력 매핑을 얻기 위해 선형 매핑

$$W_{final} d_{out}$$

을 적용.

Interpretability

TabNet의 features selection masks는 각 단계에서 선택한 features을 밝힐 수 있음. 만약

$$M_{b,j}[i] = 0$$

이면, b 번째 샘플의 j 번째 특성은 결정에 영향을 주지 않아야합니다.

f_i가 선형 함수인 경우, 계수 M_b,j[i]는 f_b,j의 특성 중요도에 해당합니다.

- 각 결정 단계는 비선형 처리를 사용하지만 출력은 나중에 선형 방식으로 결합됩니다.
- 각 단계의 분석 외에도 집계된 features 중요도를 정량화하는 것을 목표로 함.

- 여러 단계에서 마스크를 결합하려면 결정에서 각 단계의 상대적 중요성을 가늠할 수 있는 계수가 필요

우리는 b번째 샘플에 대한 i번째 결정 단계에서 총 결정 기여도를 표시하기 위해 아래를 제안

$$\eta_b[i] = \sum_{c=1}^{N_d} ReLU(d_{b,c}[i])$$

직관적으로

$$d_{b,c}[i] < 0$$

인 경우 결정 단계의 모든 features이 전체 결정에 0으로 기여해야합니다.

- 값이 증가하면 전체 선형 조합에서 더 높은 역할을 함.
- 각 의사 결정 단계에서 의사 결정 마스크를 스케일링하여 집계 특성 중요도 마스크

$$M_{agg-b,j} = \sum_{i=1}^{N_{steps}} \eta_b[i] M_{b,j}[i] / \sum_{j=1}^{D} \sum_{i=1}^{N_steps} \eta_b[i] M_{b,j}[i]$$

를 제안

Tabular self-supervised learning

- TabNet 인코딩 표현에서 tabular features을 재구성하기위한 디코더 아키텍처를 제안
- 디코더는 각 결정 단계에서 FC 플레이어가 뒤 따르는 features transformers로 구성
- 다른 것에서 누락된 특성 열을 예측하는 작업을 제안
- 이진 마스크

$$S \in \{0,1\}^{B \times D}$$

- TabNet 인코더

$$(1 - S) \cdot \hat{f}$$

를 입력하고 디코더는 재구성 된 features

$$S \cdot \hat{f}$$

를 출력

- 모델이 알려진 features 만 강조하도록 인코더에서 P[0] = (1-S) 를 초기화하고 인코더 마지막 FC 레이어에 S를 곱하여 알려지지 않은 features을 출력
- self-supervised phase에서 reconstruction loss 를 고려

$$\sum_{b=1}^{B}\sum_{j=1}^{D}|\frac{(\hat{f}_{b,j}-f_{b,j})\cdot S_{b,j}}{\sqrt{\sum_{b=1}^{B}(f_{b,j}-1/B\sum_{b=1}^{B}f_{b,j})^2}}|^2$$

- ground truth의 모집단 표준 편차를 사용한 정규화는 features의 범위가 다를 수 있으므로 유익
- 반복할 때마다 매개 변수 p_s를 사용하여 Bernoulli 분포에서 독립적으로 S_b,j를 샘플링합니다.

Experiments

- 발표된 벤치 마크를 사용하여 회귀 또는 분류 작업을 포함하는 광범위한 문제에서 TabNet을 연구
- 모든 데이터 세트에 대해 범주 형 입력은 학습 가능한 임베딩을 사용하여 학습 가능한 1 차원 스칼라에 매핑되고 숫자 열은 전처리없이 입력
- 표준 분류 (softmax 교차 엔트로피)와 회귀 (평균 제곱 오차) 손실 함수를 사용하고 수렴 할 때까지 훈련
- TabNet 모델의 하이퍼 파라미터는 검증 세트에 최적화되어 있으며 부록에 나열되어 있음
- TabNet 성능은 ablation studies in Appendix 에서 볼 수 있듯이 대부분의 하이퍼 파라미터에 그다지 민감하지 않음
- 부록에서는 주요 하이퍼 파라미터 선택에 대한 다양한 설계 및 지침에 대한 ablation studies 도 제공
- 우리가 인용하는 모든 실험에 대해 원래 작업과 동일한 훈련, 검증 및 테스트 데이터를 사용
- Adam 최적화 알고리즘 (Kingma 및 Ba 2014) 및 Glorot 균일 초기화는 모든 모델의 학습에 사용

Instance-wise feature selection

- 두드러진 특징을 선택하는 것은 고성능, 특히 소규모 데이터 세트의 경우 중요합니다. (Chen et al. 2018)(10k 훈련 샘플로 구성됨). 6 개의 표 형식 데이터 세트를 고려합니다.
- 데이터 세트는 features의 하위 집합만 출력을 결정하는 방식으로 구성됨.
- Syn 1- Syn3의 경우 두드러진 features은 모든 인스턴스에서 동일하며 (예 : Syn2의 출력은 features X3-X6에 따라 다름), 두드러진 features이 알려진 것처럼 전역 features 선택은 높은 성능을 제공
- Syn4-Syn6의 경우 두드러진 features은 인스턴스에 따라 다르므로 (예 : Syn4의 경우 출력은 X 11의 값에 따라 X1-X2 또는 X 3-X 6에 따라 달라짐) 전역 features 선택이 차선책이 됨
- 표 1은 TabNet이 다른 것 (Tree Ensembles (Geurts, Ernst 및 Wehenkel 2006), LASSO 정규화, L2X (Chen et al. 2018))보다 성능이 뛰어나며 INVASE (Yoon, Jordon 및 van der Schaar 2019)와 동등하다는 것을 보여줍니다.

Model	*Test AUC*					
	Syn1	Syn2	Syn3	Syn4	Syn5	Syn6
No selection	.578 ± .004	.789 ± .003	.854 ± .004	.558 ± .021	.662 ± .013	.692 ± .015
Tree	.574 ± .101	.872 ± .003	.899 ± .001	.684 ± .017	.741 ± .004	.771 ± .031
Lasso-regularized	.498 ± .006	.555 ± .061	.886 ± .003	.512 ± .031	.691 ± .024	.727 ± .025
L2X	.498 ± .005	.823 ± .029	.862 ± .009	.678 ± .024	.709 ± .008	.827 ± .017
INVASE	**.690 ± .006**	.877 ± .003	**.902 ± .003**	**.787 ± .004**	.784 ± .005	.877 ± .003
Global	.686 ± .005	.873 ± .003	.900 ± .003	.774 ± .006	.784 ± .005	.858 ± .004
TabNet	.682 ± .005	**.892 ± .004**	.897 ± .003	.776 ± .017	**.789 ± .009**	**.878 ± .004**

Performance on real-world datasets

Q & A

> 💡 1. M_agg-b,j 에서 eta_b[i] i가 bold체로 되어있는데 이게 벡터? scalar?