Random Forests &
TabNet: Attentive Interpretable Tabular Learning

Nov 27, 2023

Seungeun Lee

Miscellaneous

- Authors of LightGBM: A Highly Efficient Gradient Boosting Decision Tree (NIPS'17)
- Microsoft Research Asia
- Guolin Ke (cs major), Qi Meng (math major), Thomas Finley (cs, math major), Taifeng Wang, Wei Chen (stat major), Weidong Ma, Qiwei Ye, Tie-Yan Liu (ee major)

TabNet

① | Random Forests

② | Introduction

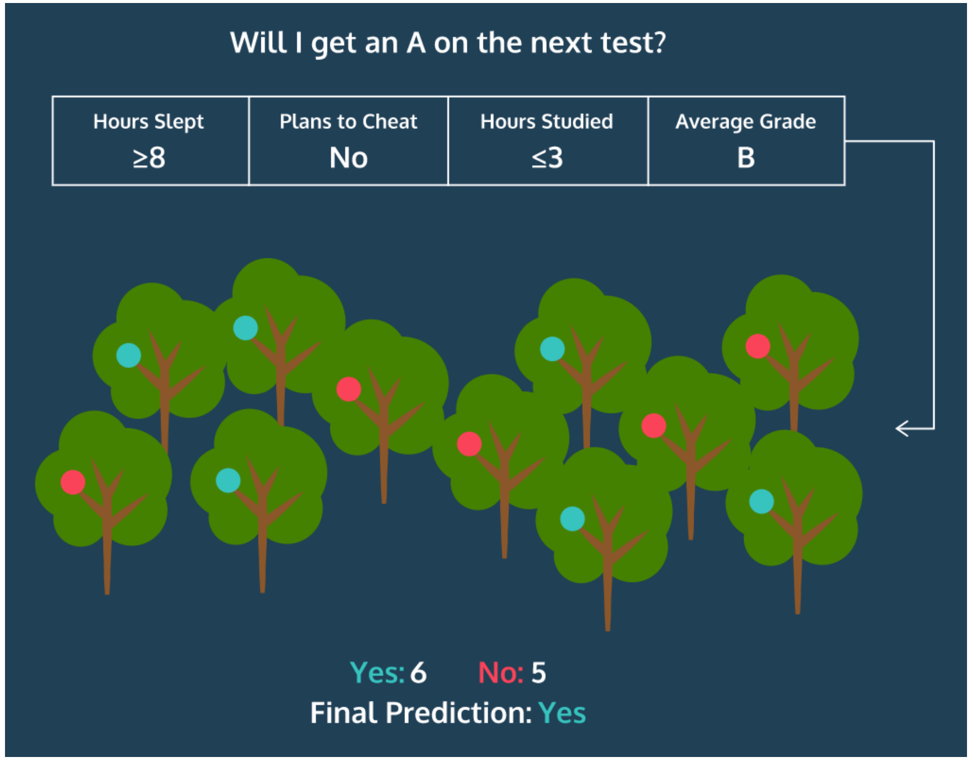
③ | Feature selection & processing

④ | Interpretability

⑤ | Details

Random Forests

- Decision Trees -> Random Forests



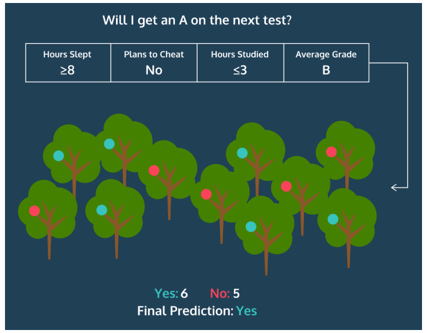

```
>>> from sklearn.ensemble import RandomForestClassifier
>>> from sklearn.datasets import make_classification
>>> X, y = make_classification(n_samples=1000, n_features=4,
...                            n_informative=2, n_redundant=0,
...                            random_state=0, shuffle=False)
>>> clf = RandomForestClassifier(max_depth=2, random_state=0)
>>> clf.fit(X, y)
RandomForestClassifier(...)
>>> print(clf.predict([[0, 0, 0, 0]]))
[1]
```

https://hleecaster.com/ml-random-forest-concept/

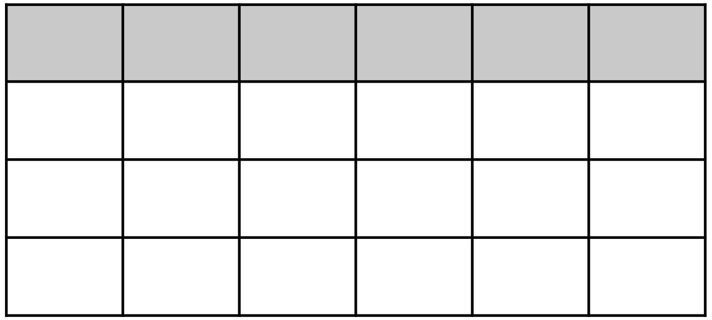
Random Forests

- Decision Trees -> Random Forests
- **Bagging ⊂ Ensemble**

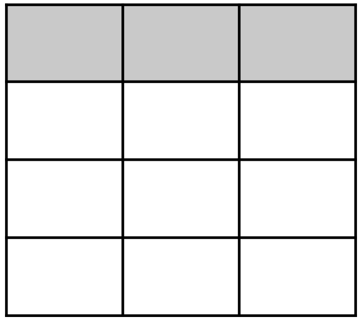
Random Forests



columns # = 10000



columns # = 100



Decision Tree

https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.etsy.com%2Flisting%2F1334300695%2Fcute-tree-svg-png-cricut-files-clipart&psig=AOvVaw1CQuXav8aA-EmN_zg6bJWK&ust=1701004588101000&source=images&cd=vfe&opi=89978449&ved=0CBEQjRxqFwoTCPCg54Oe34IDFQAAAAAdAAAAABAE

choose 100 random columns

allow duplicates

Me again!

https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.ikea.com%2Fus%2Fen%2Fp%2Fspikrak-shopping-bag-cotton-natural-70513365%2F&psig=AOvVaw0bqylJv2U54A3ZHdOEshlf&ust=1701004563937000&source=images&cd=vfe&opi=89978449&ved=0CBEQjRxqFwoTCMjor_id34IDFQAAAAAdAAAAABAE

Random Forests

- Decision Trees -> Random Forests
- Only consider subsets of columns
- How many columns to choose?
- Square root of the # of columns (empirical results)

TabNet

Introduction

- Tabular data: the most common data type in real-world Ai *(Chui et al., 2018)*
- Deep Learning for tabular data remains under-explored – still now!
- (1) DT-based approaches have certain benefits (efficiency & interpretability)
- (2) DNN not being well-suited for tabular data – overparametrized
- Why do tree-based models still outperform deep learning on tabular data? (https://arxiv.org/abs/2207.08815)

Introduction

- But why is DL worth exploring for tabular data?
- (1) expect performance improvements particularly for large datasets
- (2) gradient descent-based E2E learning (multi-modal analysis, alleviate the need for feature engineering (normalization, scaling), streaming data, E2E, Generative modeling)

Introduction

🤔 But Really?

<TabNet Contribution>

- Input: “raw” tabular data
- Sequential attention: instance-wise feature selection & interpretability
- Unsupervised learning (generative modeling)

- Sparse instance-wise feature selection
- Sequential multi-step architecture
- Improve learning capacity
- Mimic ensembling via higher dimensions and more steps

TabNet

① | Random Forests

② | Introduction

③ | Feature selection & processing

④ | Interpretability

⑤ | Details

Feature selection

(1) Raw numerical features &

trainable embeddings of categorical features (via torch.nn.Embedding – not well explained in the paper, refer to the code @github)

-> Need to specify which features are categorical variables (implementation)

TabNet does not use global normalization (rather, Batch Normalization)

(2) Encoding: sequential multi-step processing

Feature selection

- sequential multi-step (N_{steps}) processing
- Extracted information from (i-1)th step -> ith step
- Inspired by the structure of machine learning algorithms using Decision Trees w/ boosting (e.g. XGBoost, LightGBM)

Feature selection

- sequential multi-step (N_{steps}) processing

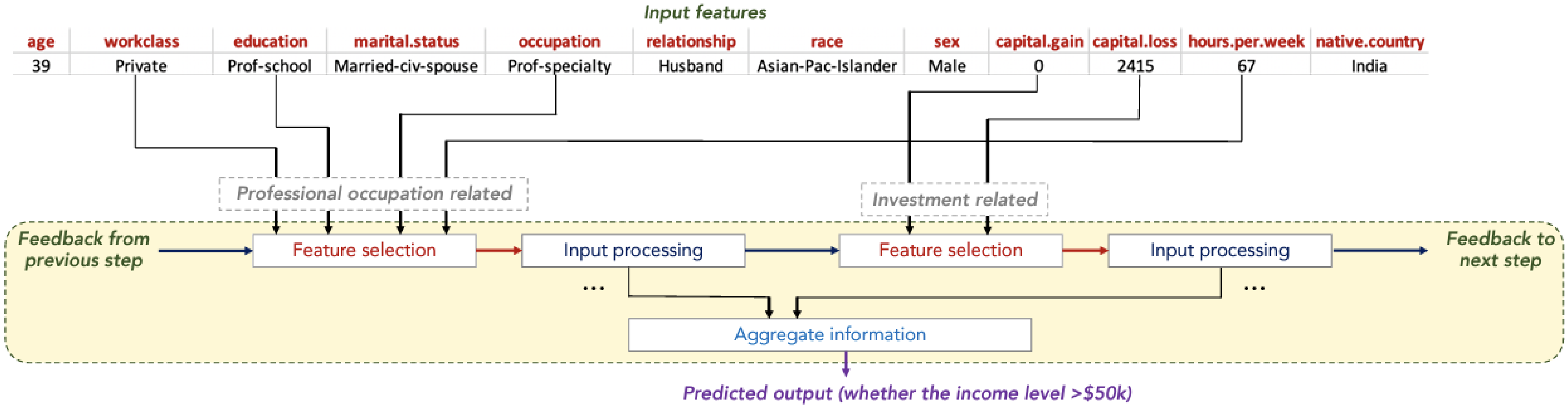


Figure 1: TabNet's sparse feature selection exemplified for Adult Census Income prediction (Dua and Graff 2017). Sparse feature selection enables interpretability and better learning as the capacity is used for the most salient features. TabNet employs multiple decision blocks that focus on processing a subset of input features for reasoning. Two decision blocks shown as examples process features that are related to professional occupation and investments, respectively, in order to predict the income level.

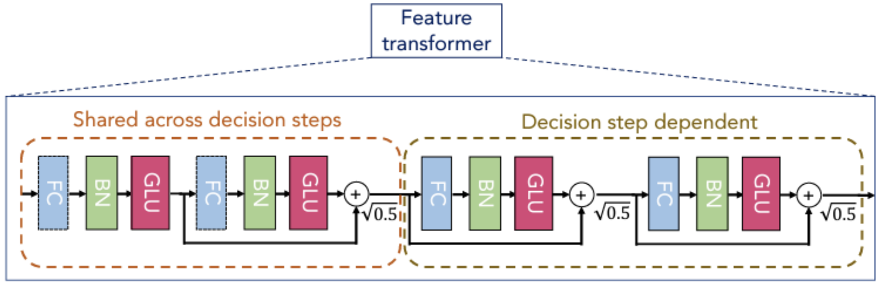
Feature selection

Split block: divides the processed representation to be used by the attentive transformer of the subsequent step as well as for the overall output

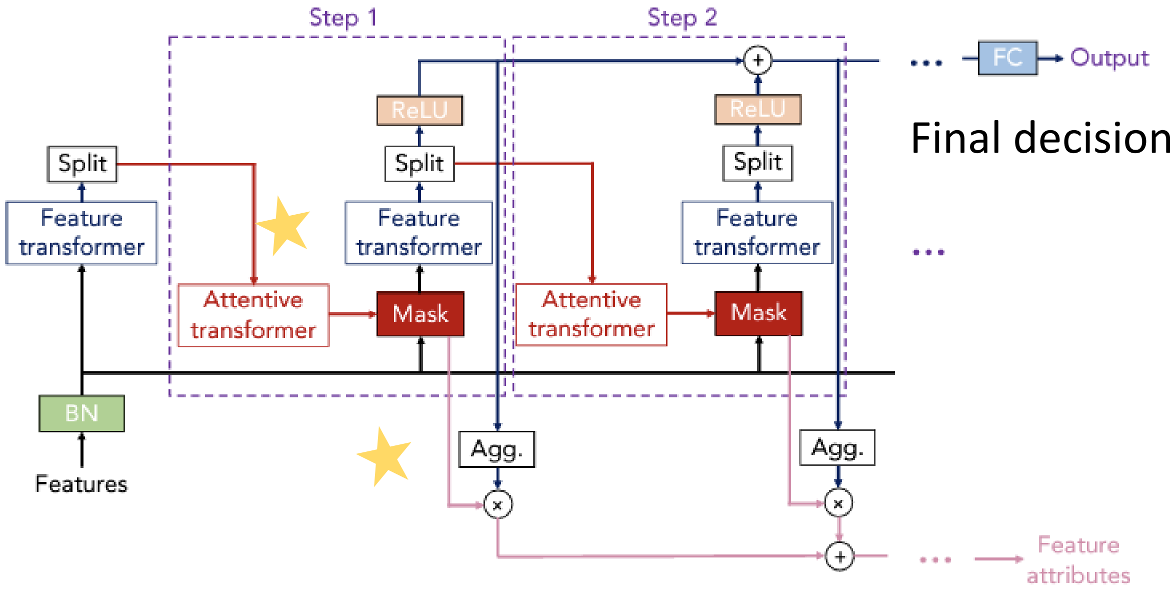
Note. Not “that” transformer

- sequential multi-step (N_{steps}) processing

regard the process w/ 2nd feature transformer as step 1



(c)



Final decision

(a) TabNet encoder architecture

Interpretability information

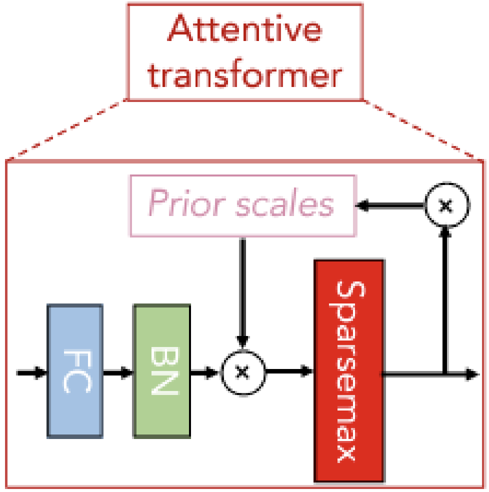
Feature selection

- Learnable mask $M[i] \in \mathbb{R}^{n \times d}$ for soft selection of the salient features
- can be viewed as a sparse selection & advantage of being effective
- The masking is multiplicative i.e. $M[i] \cdot f$
- $M[i] = sparsemax(P[i-1] \cdot h_i(a[i-1])),$

where $\prod_{j=1}^{D} M[i]_{b,j} = 1$ (jth feature of bth sample)

-> <u>instance-wise feature selection</u>

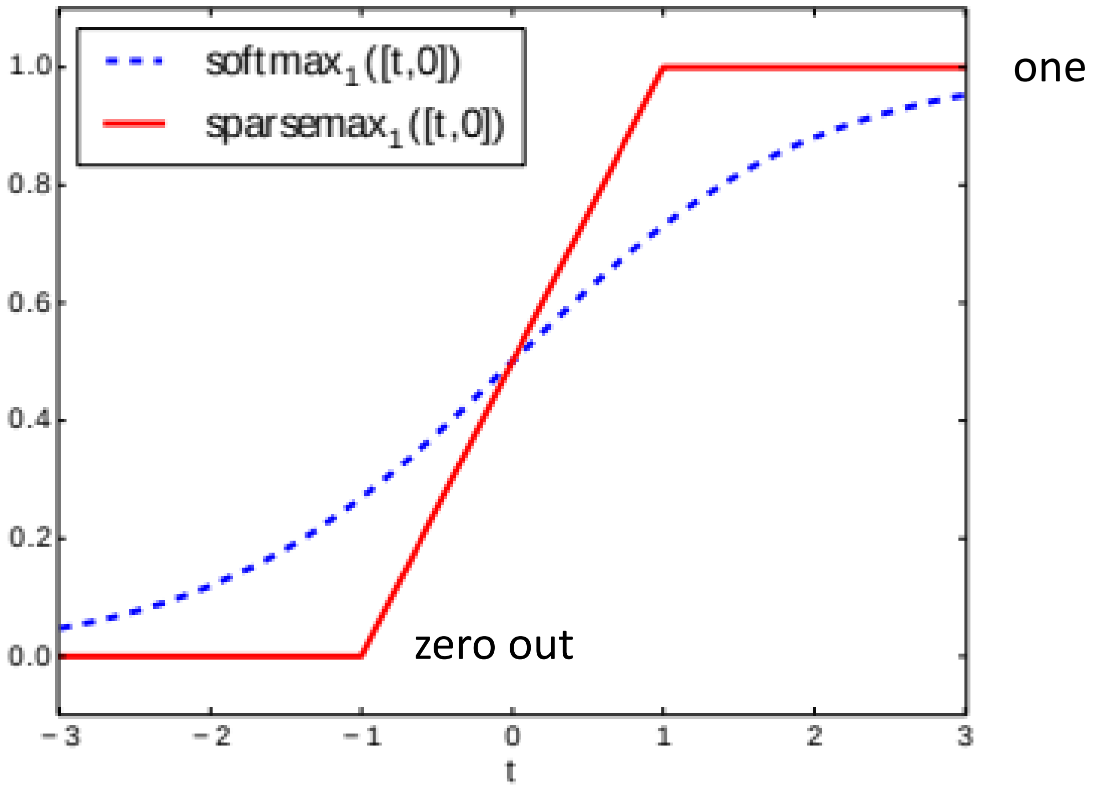
- Here, h_i is a trainable function with FC + BN
- $a[i-1]$ is from the previous step
- $P[i] = \prod_{j=1}^{i} (\gamma - M[j])$: prior scale term



(d)

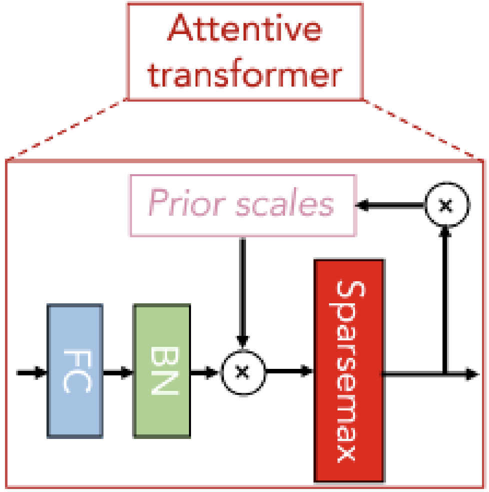
Feature selection

- Sparsemax *(Martins and Austdillo, 2016)*
- Encourages sparsity by mapping the Euclidean projection onto the probabilistic simplex
- Superior in performance and aligned with the goal of sparse feature selection for explainability (also for effectiveness)



https://www.google.com/url?sa=i&url=https%3A%2F%2Ftowardsdatascience.com%2Fwhat-is-sparsemax-f84c136624e4&psig=AOvVaw0RUgCi-ntNQ3sYtlS2fq-B&ust=1701071057495000&source=images&cd=vfe&opi=89978449&ved=0CBEQjRxqFwoTCJCF-dKV4YIDFQAAAAAdAAAAABAE

Feature selection



(d)

- $M[i] = sparsemax(P[i-1] \cdot h_i(a[i-1])),$
- $P[i] = \prod_{j=1}^{i} (\gamma - M[j])$: prior scale term
- How much a particular feature has been used previously
- Initialization: $P[0]$ as all ones (w.o. any prior on the masked features)
- γ: relaxation parameter
- If $\gamma = 1$, a feature is enforced to be used only at one decision step
- Else if γ increases, more flexibility is provided to use a feature at multiple decision steps

Feature selection

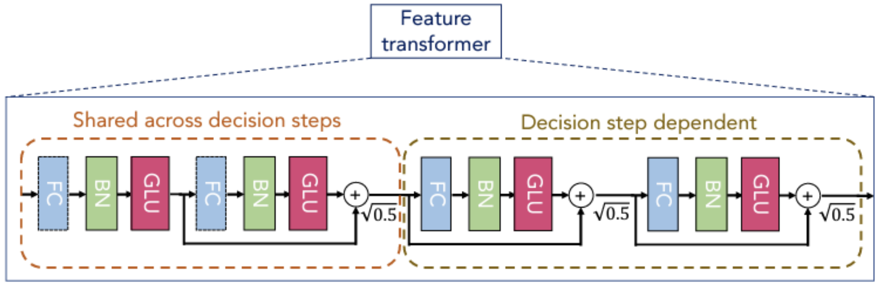
- Add the sparsity regularization to the overall loss, with a coefficient $\lambda_{coefficient}$
- Sparsity regularization in the form of entropy *(Grandvalet and Bengio, 2004)*

↑ more masks

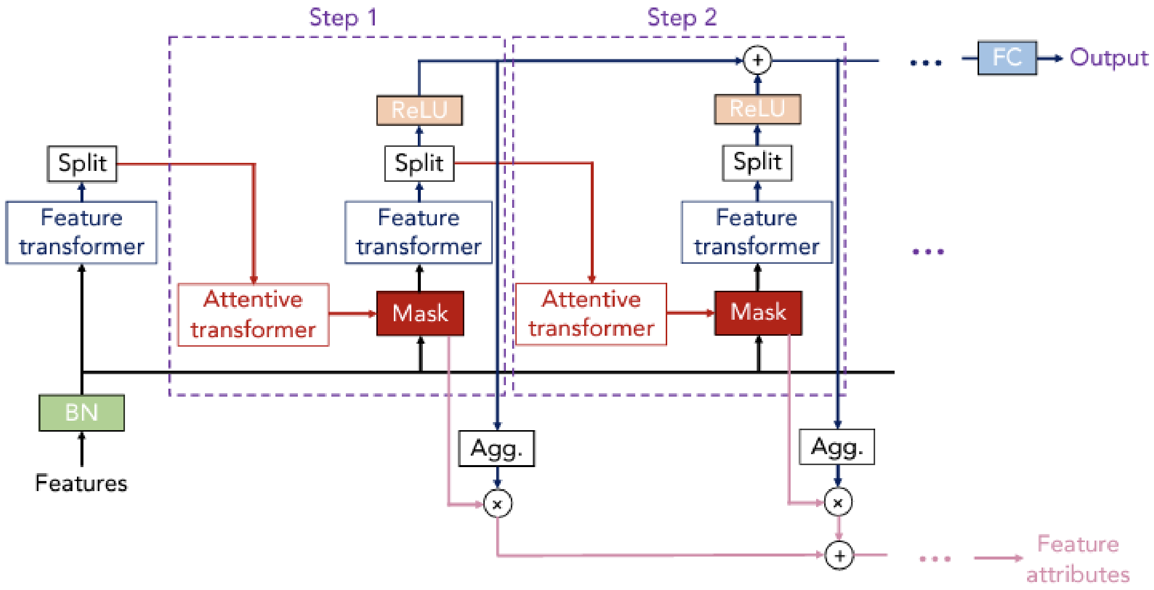
$$L_{sparse} = \sum_{i=1}^{N_{steps}} \sum_{b=1}^{B} \sum_{j=1}^{D} \frac{-\mathbf{M_{b,j}}[\mathbf{i}] \log(\mathbf{M_{b,j}}[\mathbf{i}]+\epsilon)}{N_{steps} \cdot B} \downarrow$$

- ϵ: a small number for numerical stability
- Sparsity provides a favorable inductive bias for datasets where most features are redundant

Feature processing



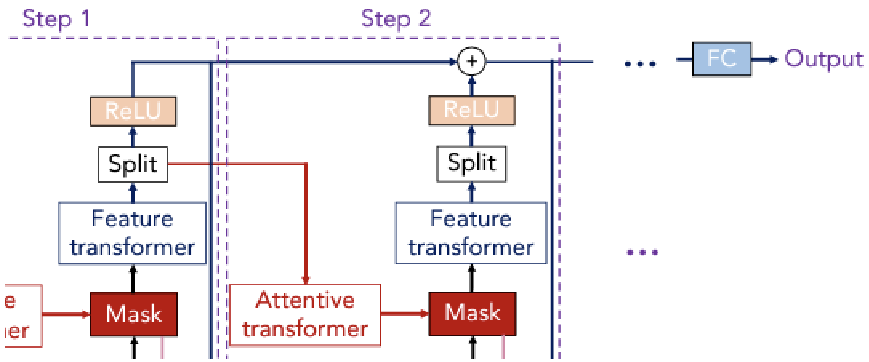
(c)



(a) TabNet encoder architecture

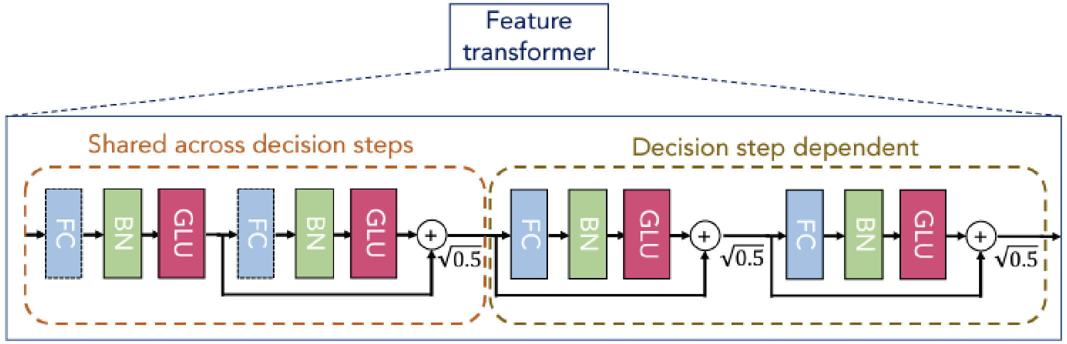
Feature processing

- $[d[i], a[i]] = f_i(M[i] \cdot f)$, where $d[i] \in \mathbb{R}^{B \times N_d}$ and a$[i] \in \mathbb{R}^{B \times N_a}$
- $d[i]$ -> for final decision
- a$[i]$ -> ith step value (mask)



Feature processing

- 2 layers are shared across all decision steps (share weights)
- 2 are decision step-independent



(c)

- FC, BN, GLU
- Normalized residual connection w/ $\sqrt{0.5}$

Feature processing

- Ghost BN *(Hoffer, Hubara, and Soudry, 2017) – virtual batch?*

- $d_{out} = \sum_{i=1}^{N_{steps}} ReLU(d[i])$
- Final linear mapping $W_{final} d_{out}$

TabNet

① | Random Forests

② | Introduction

③ | Feature selection & processing

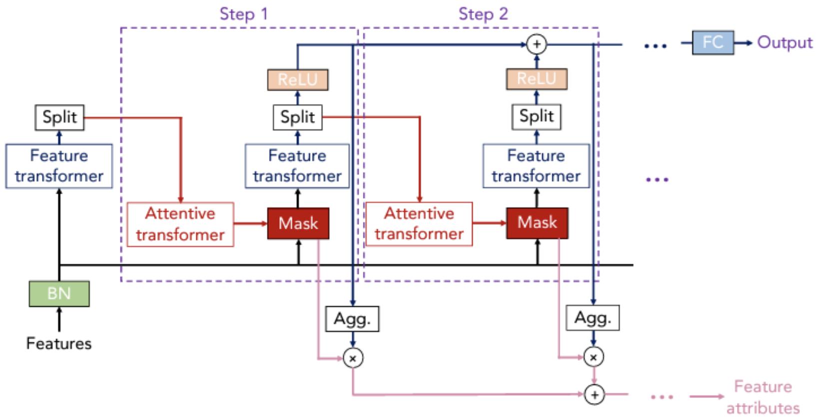
④ | Interpretability

⑤ | Details

Interpretability

- $M[i]_{b,j} = 0$ -> jth feature of bth sample should have no contribution to the decision
- Each decision step employs non-linear (DL) processing, their outputs are combined later in a linear way (aggregator)
- Aim to quantify an aggregate feature importance in addition to analysis of each step

Interpretability

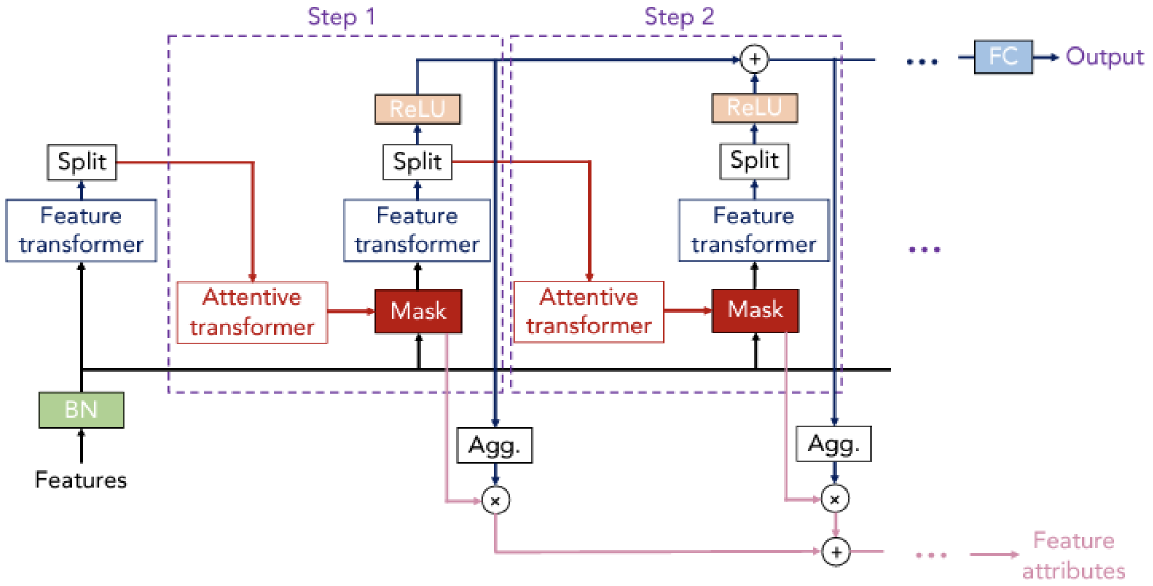


(a) TabNet encoder architecture

- $\eta_b[i] = \sum_{c=1}^{N_d} ReLU(d_{b,c}[i])$
- Aggregate decision contribution at ith decision step for the bth sample
- Intuitively, $d_{b,c}[i] < 0$, no role / $d_{b,c}[i]$ increases, plays a higher role

- <u>Recap</u>
- $[d[i], a[i]] = f_i(M[i] \cdot f)$, where $d[i] \in \mathbb{R}^{B \times N_d}$ and a$[i] \in \mathbb{R}^{B \times N_a}$
- $d[i]$ -> for final decision
- a$[i]$ -> ith step value (mask)

Interpretability



(a) TabNet encoder architecture

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

jth feature, bth sample

Interpretability

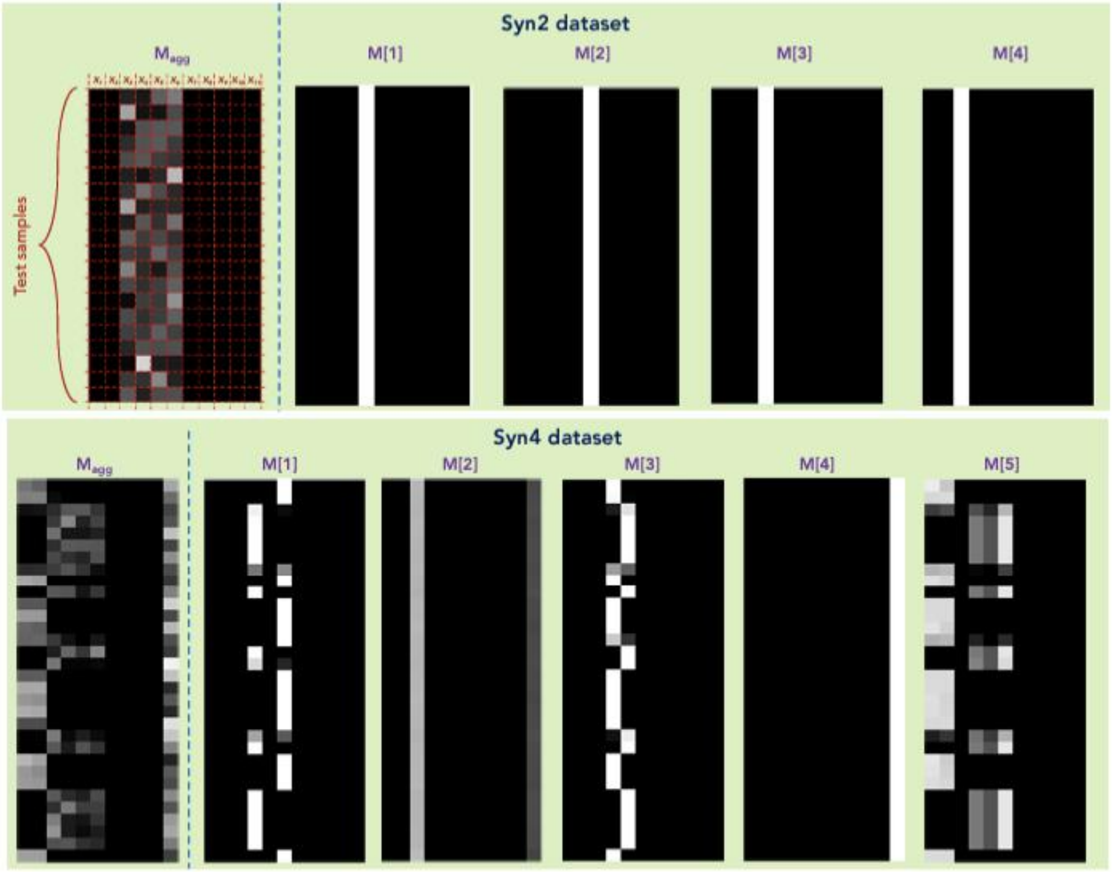
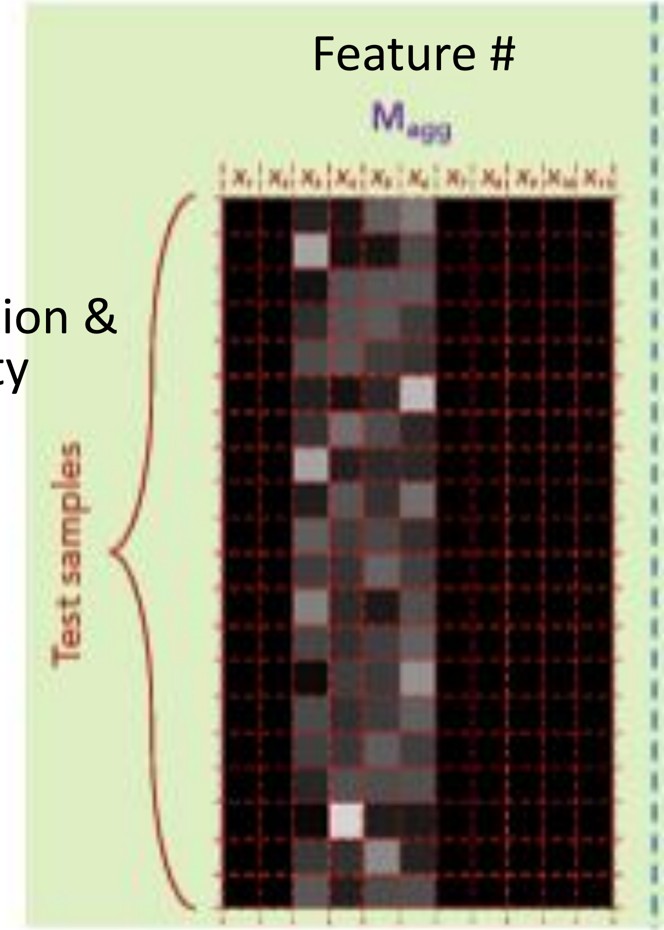


Figure 5: Feature importance masks $\mathbf{M}[\mathbf{i}]$ (that indicate feature selection at i^{th} step) and the aggregate feature importance mask $\mathbf{M_{agg}}$ showing the global instance-wise feature selection, on Syn2 and Syn4 (Chen et al. 2018). Brighter colors show a higher value. E.g. for Syn2, only X_3-X_6 are used.

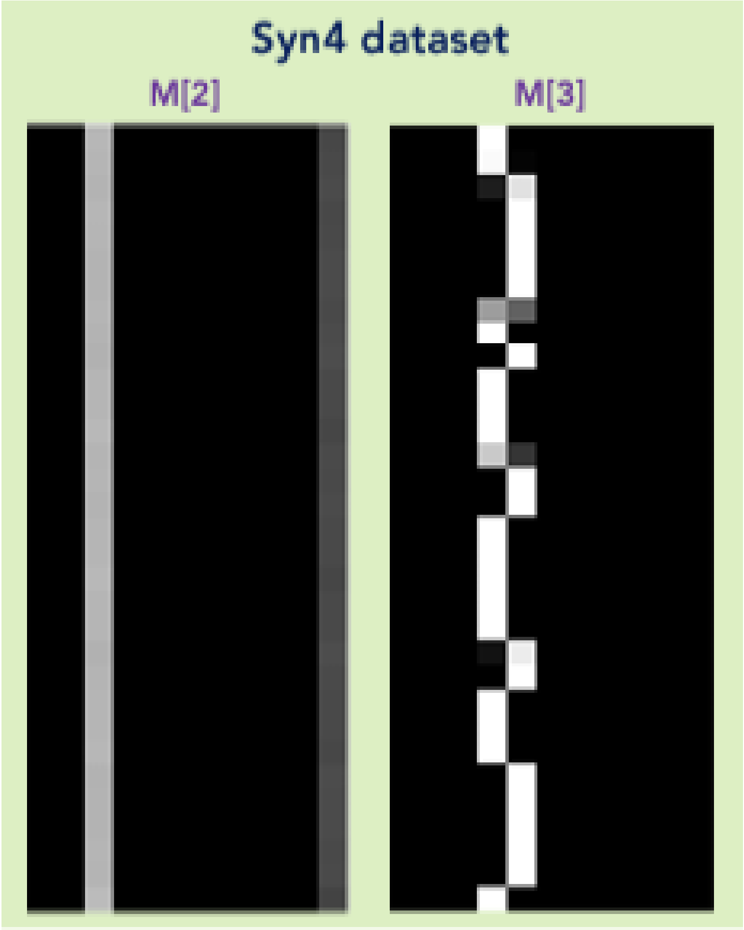
Interpretability

Feature selection of ith step
(here, 2nd 3rd)

Instance-wise
feature selection &
interpretability



Brighter color,
higher contribution



TabNet

① | Random Forests

② | Introduction

③ | Feature selection & processing

④ | Interpretability

⑤ | Details

Details

Performance on real-world datasets

Table 2: Performance for Forest Cover Type dataset.

Model	*Test accuracy (%)*
XGBoost	89.34
LightGBM	89.28
CatBoost	85.14
AutoML Tables	94.95
TabNet	**96.99**

Forest Cover Type (Dua and Graff 2017): The task is classification of forest cover type from cartographic variables. Table 2 shows that TabNet outperforms ensemble tree based approaches that are known to achieve solid performance (Mitchell et al. 2018). We also consider AutoML Tables (AutoML 2019), an automated search framework based on ensemble of models including DNN, gradient boosted DT, AdaNet (Cortes et al. 2016) and ensembles (AutoML 2019) with very thorough hyperparameter search. A single TabNet without fine-grained hyperparameter search outperforms it.

Table 3: Performance for Poker Hand induction dataset.

Model	*Test accuracy (%)*
DT	50.0
MLP	50.0
Deep neural DT	65.1
XGBoost	71.1
LightGBM	70.0
CatBoost	66.6
TabNet	**99.2**
Rule-based	100.0

Poker Hand (Dua and Graff 2017): The task is classification of the poker hand from the raw suit and rank attributes of the cards. The input-output relationship is deterministic and hand-crafted rules can get 100% accuracy. Yet, conventional DNNs, DTs, and even their hybrid variant of deep neural DTs (Yang, Morillo, and Hospedales 2018) severely suffer from the imbalanced data and cannot learn the required sorting and ranking operations (Yang, Morillo, and Hospedales 2018). Tuned XGBoost, CatBoost, and LightGBM show very slight improvements over them. TabNet outperforms other methods, as it can perform highly-nonlinear processing with its depth, without overfitting thanks to instance-wise feature selection.

Details

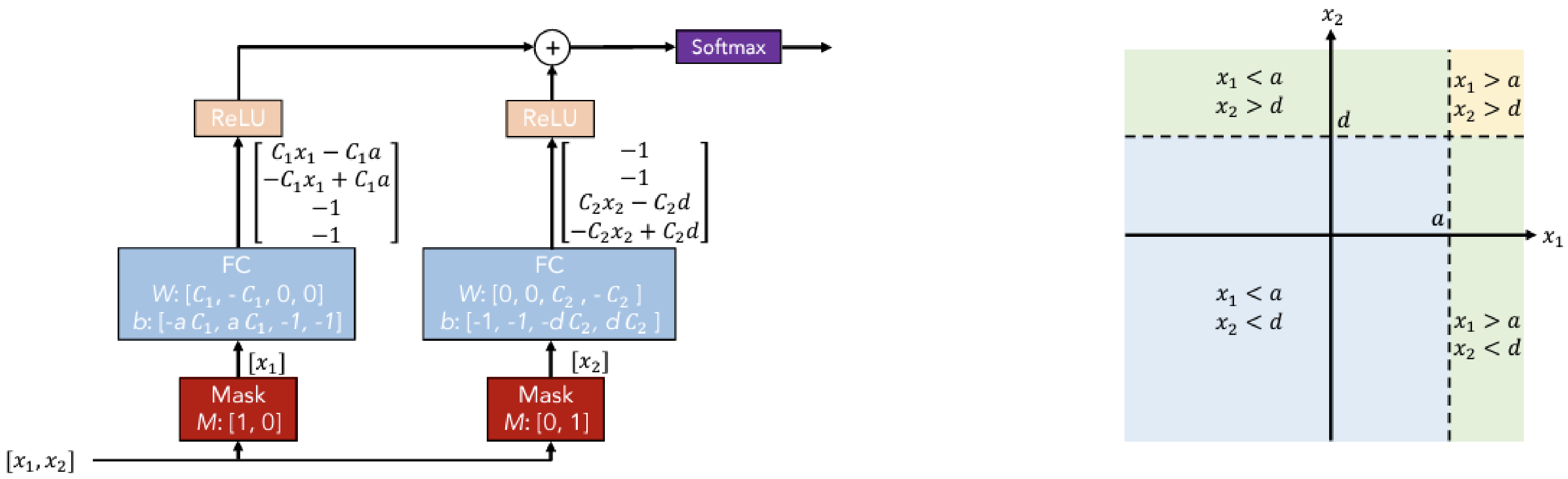
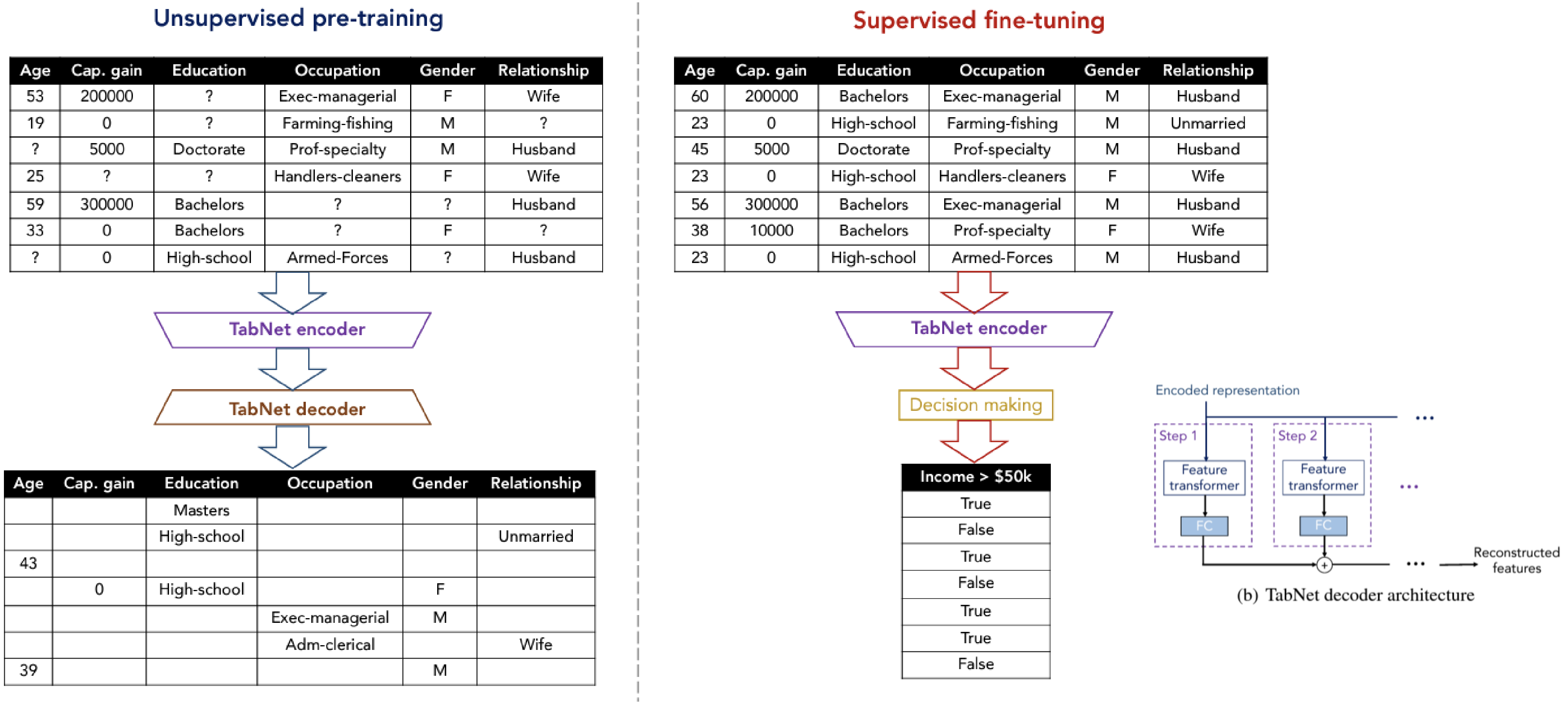


Figure 3: Illustration of DT-like classification using conventional DNN blocks (left) and the corresponding decision manifold (right). Relevant features are selected by using multiplicative sparse masks on inputs. The selected features are linearly transformed, and after a bias addition (to represent boundaries) ReLU performs region selection by zeroing the regions. Aggregation of multiple regions is based on addition. As C_1 and C_2 get larger, the decision boundary gets sharper.

Details



Age	Cap. gain	Education	Occupation	Gender	Relationship
53	200000	?	Exec-managerial	F	Wife
19	0	?	Farming-fishing	M	?
?	5000	Doctorate	Prof-specialty	M	Husband
25	?	?	Handlers-cleaners	F	Wife
59	300000	Bachelors	?	?	Husband
33	0	Bachelors	?	F	?
?	0	High-school	Armed-Forces	?	Husband

Age	Cap. gain	Education	Occupation	Gender	Relationship
		Masters			
		High-school			Unmarried
43					
	0	High-school		F	
			Exec-managerial	M	
			Adm-clerical		Wife
39				M	

Age	Cap. gain	Education	Occupation	Gender	Relationship
60	200000	Bachelors	Exec-managerial	M	Husband
23	0	High-school	Farming-fishing	M	Unmarried
45	5000	Doctorate	Prof-specialty	M	Husband
23	0	High-school	Handlers-cleaners	F	Wife
56	300000	Bachelors	Exec-managerial	M	Husband
38	10000	Bachelors	Prof-specialty	F	Wife
23	0	High-school	Armed-Forces	M	Husband

Income > $50k
True
False
True
False
True
True
False

Figure 2: Self-supervised tabular learning. Real-world tabular datasets have interdependent feature columns, e.g., the education level can be guessed from the occupation, or the gender can be guessed from the relationship. Unsupervised representation learning by masked self-supervised learning results in an improved encoder model for the supervised learning task.

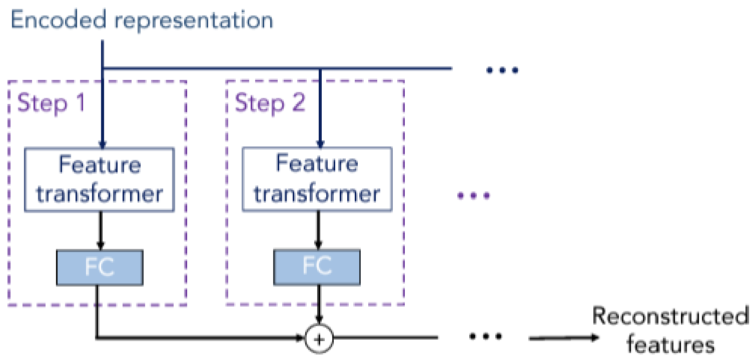
Details

- Reconstruct TabNet encoded representations
- Feature transformer + FC -> output sum
- Binary mask $S \in \{0,1\}^{B \times D}$
  - sample $S_{b,j}$ from a Bernoulli distribution w/ different parameter at each iteration
- Encoder inputs: $(1-S) \cdot \hat{f}$ (initialize $\mathrm{P}[0] = 1 - S$)
- Decoder outputs: $S \cdot \hat{f}$ (initialization: multiply S)
- Reconstruction loss for self-supervised learning

Same mask, similar value

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

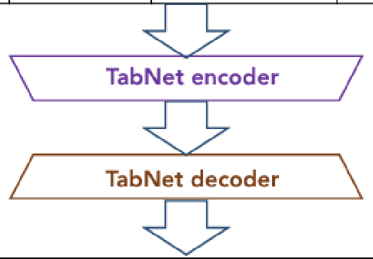
- Normalization was beneficial (only in this case)



(b) TabNet decoder architecture

Unsupervised pre-training

Age	Cap. gain	Education	Occupation	Gender	Relationship
53	200000	?	Exec-managerial	F	Wife
19	0	?	Farming-fishing	M	?
?	5000	Doctorate	Prof-specialty	M	Husband
25	?	?	Handlers-cleaners	F	Wife
59	300000	Bachelors	?	?	Husband
33	0	Bachelors	?	F	?
?	0	High-school	Armed-Forces	?	Husband



Age	Cap. gain	Education	Occupation	Gender	Relationship
		Masters			
		High-school			Unmarried
43					
	0	High-school		F	
			Exec-managerial	M	
			Adm-clerical		Wife
39				M	

Details

- https://github.com/dreamquark-ai/tabnet
- Code!

References

- https://arxiv.org/abs/1908.07442
- Many thanks to https://www.intelligencelabs.tech/3ac72939-db45-4804-9b9d-3ec2c08ef504

What's Next?

- Meta Learning
- Deep Tabular Learning w/ Meta Learning (STUNT)