**Tree-based Models vs. Neural Networks for Tabular Data**

**Project 1: Summary of Findings**

Abou Keita and William Simpson

Dept. of Computer Science & Electrical Engineering, University of Maryland, Baltimore County

DATA 690: Practical Deep Learning

Dr. Weizhe Li

October 12, 2022

**Tree-based Models vs. Neural Networks for Tabular Data**

Despite advances in deep learning neural networks that allow them to excel at computer vision and natural language processing (NLP) tasks, neural network performance on tabular data still generally lags behind that of tree-base models. In this project, we construct Random Forest and Multilayer Perceptron (MLP) models to predict housing prices using a historical dataset of houses in Ames, Iowa. Our Random Forest model achieves an R2 score of 0.88, while the NN achieves an R2 of 0.75. This aligns with existing research that tree-based models tend to outperform Neural Networks (NNs) on tabular data. Literature commonly attributes this performance gap to: 1) properties of the data itself, and 2) properties of the model. Researchers also propose methods to reduce this gap. The summary below is a synthesis of this literature.

**1. Properties of the Data**

First regarding the data, tabular data is typically **heterogeneous** being composed of many different feature types like dense numerical and sparse categorical features (Borisov et al., 2021). Image, text, and audio data that NNs excel at processing are homogenous, with all features being of one type (e.g., pixels) (Grinsztajn, Oyallon, & Varoquaux, 2022). The heterogeneity of tabular data means there are many **uninformative features** which affect MLP-like NNs more than tree models (Grinsztajn, Oyallon, & Varoquaux, 2022). Removing uninformative features through feature selection can reduce the performance gap (Grinsztajn, Oyallon, & Varoquaux, 2022). However, this leaves a second issue stemming from the properties of tabular data— the **importance of single features**. In tabular data, variations in a single feature can substantially change the final prediction (Borisov et al., 2021). Tree models handle this well, but NNs struggle with this since they are better suited for coordinated changes in homogenous features, such as a collection of pixels in a particular region of an image.

Tabular data exhibits a third difficulty for NNs pertaining to the **quality of training data** (Borisov et al., 2021). Tabular data often has many missing values, outliers, erroneous and inconsistent data, as well as a small amount of records relative to a large number of features. Tree based models can deal with these irregularities internally, but NNs expect more data consistency and without it their performance suffers. Fourth, there are often **no (or irregular) dependencies between features in tabular data** (Borisov et al., 2021). While tree-based models take advantage of this by relying on relative feature importances in their prediction, NNs such as Convolutional NNs actually expect to leverage spatial relations between features. Finally, the **amount of data** impacts performance of NNs compared to tree-based models. In a study proposing a self-normalizing neural network architecture, Klambauer, Unterthiner, & Mayr (2017) found that Random Forest (and Support Vector Machine) outperformed NNs on 75 small tabular datasets of fewer than 1000 rows. However, for larger datasets of more than 1000 rows, the NNs achieved higher performance (Klambauer, Unterthiner, & Mayr, 2017). This suggests there are some instances in analyzing tabular data where NNs may be preferable, but researchers note that underlying properties of neural models restrict such use cases.

**2. Properties of the Model**

Machine learning algorithms, including NNs, make implicit or explicit assumptions (i.e., inductive biases) in order to generalize from a training dataset. These impact how a model performs in learning from a tabular dataset. NNs are intentionally designed to exhibit **non-linearity and high complexity** because these properties of NNs are essential for computer vision and NLP. However, the same characteristics can also be the reason that NNs perform poorly on simpler and smaller tabular datasets (Borisov et al., 2021). Another quality of NNs in this domain is that they are **biased to produce excessively smooth solutions** (Grinsztajn, Oyallon, & Varoquaux, 2022). In other words, the target function in machine learning for tabular data is typically not smooth. So, while NNs allow non-linearity, they must also be differentiable. Tree models are better able to fit the irregular functions that appear in tabular data.

Research also finds that the performance gap between NNs and tree models on tabular data relates to the fact that an MLP-type network’s **learning procedure is invariant by rotation** (Grinsztajn, Oyallon, & Varoquaux, 2022). This means that the network’s learning is unaffected by applying rotations to the data features. Importantly, tabular data is non invariant by rotation and thus models whose learning procedure is also non invariant by rotation such as tree models are better suited. Finally, other factors such as the specific data preprocessing steps can substantially impact NN performance on tabular data (Borisov et al., 2021). One-hot encoding, for example, produces a sparse feature matrix that can cause difficulty for NNs.

**Strategies for using NNs and Tabular Data**

Research on the use of NNs for tabular data analysis also explores how to improve the performance of NNs for this purpose. Three main categories of solutions emerge in the literature. First, **data transformations** aim to process tabular data to make it better suited for NNs (Hancock & Khoshgoftaar, 2020; Borisov et al., 2021). This includes single and multiple dimensional encodings of features which help NNs better extract information signals in the data. Second, **specialized neural architectures** have been proposed to overcome structural issues of using NNs for tabular data discussed above (Shwartz-Ziv & Armon, 2022; Borisov et al., 2021). One type of architecture is a hybrid tree-neural model, also called differentiable trees, which merge classical machine learning with NNs (Shwartz-Ziv & Armon, 2022; Borisov et al., 2021). Another model type is transformer-based using mechanisms of attention (Shwartz-Ziv & Armon, 2022; Borisov et al., 2021). Lastly, substantial **regularization** has been investigated as a way to enable non-linear and complex NNs to better fit structured data (Shwartz-Ziv & Armon, 2022; Borisov et al., 2021). This is often in the form of specialized loss functions.

**Conclusions**

Overall, our model results from this project and existing literature support that tree-based models tend to achieve better performance for tabular data. Tree-based models are also faster and easier to optimize (Klambauer, Unterthiner, & Mayr, 2017). However, there are some cases when a NN can outperform a tree model on larger volumes of tabular data (Klambauer, Unterthiner, & Mayr, 2017). There is evidence that an ensemble of both tree models and NNs even achieves performance beyond either individually (Shwartz-Ziv & Armon, 2022). Therefore, deep learning NNs are still valuable and applicable tools to study for non-tabular but also tabular data.

**References**

Borisov, V., Leemann, T., Seßler, K., Haug, J., Pawelczyk, M., & Kasneci, G. (2021). Deep neural networks and tabular data: A survey. arXiv preprint arXiv:2110.01889.

Hancock, J. T., & Khoshgoftaar, T. M. (2020). Survey on categorical data for neural networks. Journal of Big Data, 7(1), 1-41.

Grinsztajn, L., Oyallon, E., & Varoquaux, G. (2022). Why do tree-based models still outperform deep learning on tabular data?. arXiv preprint arXiv:2207.08815.

Klambauer, G., Unterthiner, T., Mayr, A., & Hochreiter, S. (2017). Self-normalizing neural networks. Advances in neural information processing systems, 30.

Shwartz-Ziv, R., & Armon, A. (2022). Tabular data: Deep learning is not all you need. Information Fusion, 81, 84-90.

# Research Notes

### **Topic**: Why, when do tree-based models perform better than neural networks?; How to make NN suitable for tabular data/vice versa?

**Grinsztajn et al 2022**

* “Why do tree-based models still outperform deep learning on tabular data?”
* Methods
  + Conduct benchmarks for tree-based vs NN on tab data
* Conclusion:
  + “tree- based models remain state-of-the-art on medium-sized data (∼10K samples) even without accounting for their superior speed.”
* Details
  + Explain differences in perf:
    - “neural networks struggle to learn irregular patterns of the target function, and their rotation invariance hurt their performance, in particular when handling the numerous uninformative features present in tabular data.”
    - “partial explanation: MLPs are expressive enough for tabular data but may suffer from a lack of proper regularization.”
    - WHY: Klambauer et al., 2017, Borisov et al., 2021].
  + Tuning hyperparameters does not make NNs state-of-the-art
  + Categorical variables are not the main weakness of NNs
  + inductive biases of decision trees that make them well-suited for tabular data, and how they differ from the inductive biases of NNs
    - Finding 1: NNs are biased to overly smooth solutions
      * target functions in our datasets are not smooth, and that NNs struggle to fit these irregular functions compared to tree-based models
    - Finding 2: Uninformative features affect more MLP-like NNs
      * removing uninformative features (5a) reduces the performance gap between MLPs (Resnet) and the other models (FT Transformers and tree-based models)
    - Finding 3: Data are non invariant by rotation, so should be learning procedures
      * MLPs much more hindered by uninformative features, compared to other models? One answer is that this learner is rotationally invariant in the sense of Ng [2004]: the learning procedure which learns an MLP on a training set and evaluate it on a testing set is unchanged when applying a rotation (unitary matrix) to the features on both the training and testing set.

**Borisov et al 2021**

* “Deep Neural Networks and Tabular Data: A Survey”
* From grinsztajn: attempts to make deep learning work on tabular data:
  + data encoding techniques to make tabular data better suited for deep learning [Hancock and Khoshgoftaar, 2020, Yoon et al., 2020],
  + "hybrid methods" to benefit from the flexibility of NNs while keeping the inductive biases of other algorithms like tree-based models [Lay et al., 2018, Popov et al., 2020, Abutbul et al., 2020, Hehn et al., 2019, Tanno et al., 2019, Chen, 2020, Kontschieder et al., 2015, Rodriguez et al., 2019, Popov et al., 2020, Lay et al., 2018] or
  + Factorization Machines Guo et al. [2017],
  + tabular- specific transformers architectures Somepalli et al. [2021], Kossen et al. [2021], Arik and Pfister [2019], Huang et al. [2020],
  + and various regularization techniques to adapt classical architectures to tabular data [Lounici et al., 2021, Shavitt and Segal, 2018, Kadra et al., 2021a, Fiedler, 2021]
* overview of state of the art dl methods for tabular data
  + 1 Data transformations,
    - Transform categorical and numerical data to enable NN to better extract info signal
    - Cost = inc preprocessing time
    - types
      * Single dim encodings - transform each feature independently
      * Multi dim encodings - transform entire “record to another representation”
  + 2 specialized architectures
    - Covers biggest share of works in this domain
    - Hybrid models - fuse classical ML with NN
    - transformer -based models – attention mechanisms
  + 3 regularization models
    - Purpose: reason that poor NN tab data perf is bc of NN high nonlinearity and complexity → regularize – “special purpose loss functions”
* “Tabular data - in contrast to image or language data – are heterogeneous, leading to dense numerical and sparse categorical features
* Reasons for different perf – WHY
  + Low-quality training data: missing values, outliers, erroneous/inconsistent data, small dataset size relative to high dimensional number of features
    - Tree algs can deal with these internally
  + Missing or Complex Irregular Spatial Dependencies: often no or irregular dependencies between features in tabular data and so models (NN) that are popular for homogenous data aren’t well suited to this
  + Dependency on Preprocessing: NN perf on tab data is highly dependent on the preprocessing done on the data – difficulty with sparse feature matrix (one hot encoding)
  + Importance of single features: in tab data a single feature can significantly change the final prediction –tree models handle this very well – NN don’t – better suited for when a coordinated change in many features results in a change in the final prediction
* “However, we observed that for very large data sets, approaches based on deep learning may still be able to achieve competitive performance and even outperform classical models.”

**Klambauer et al 2017**

* “Self-normalizing Neural networks”
* “On 75 small datasets with less than 1000 data points, random forests and SVMs outperform SNNs and other FNNs. On 46 larger datasets with at least 1000 data points, SNNs show the highest performance followed by SVMs and random forests

**Shwartz-Ziv and Armon 2022**

* “Tabular data: Deep learning is not all you need
* Conclusions:
  + XGBoost outperforms deep models across datasets tested
  + Ensemble of xgboost and deep models performs better than xgboost alone
* Neural arch for tab data
  + Differentiable trees
  + Attention-based models
  + Other
    - Regularization
    - Expliciting modeling of multiplicative interactions
    - 1d-cnn - take advantage of convolutions for tab data
* XGBoost - easiest to optimize and best results over nn models → ensemble for max perf