SINGLE-TASK AND MULTI-TASK XGBOOST ON TABULAR REGRESSION TASKS

Cormac Cureton

ECSE 507 Course Project
McGill University
Department of Electrical and Computer Engineering

ABSTRACT

Tabular data remains a difficult domain for the application of machine learning techniques, due to a lack of structure in the data. Although deep learning methods have been applied in recent years, gradient boosting models like XGBoost have remained competitive in many tabular scenarios. This paper investigates the performance of single-task and multi-task XGBoost models on three mid-sized tabular datasets. Our experiments yield mixed results, the multi-task models improve performance on some tasks within some datasets but no model dominates across all tasks and datasets. Additionally, we investigate the impact of standardizing features and targets but find that it did not meaningfully impact performance. These results suggest that multi-task XGBoost may offer improvements in some circumstances but it should be investigated on a case-by-case basis rather than being applied universally.

***Index Terms*—** Multi-task learning, XGBoost, Tabular Data

1. INTRODUCTION

In domain like computer vision and natural language processing, deep learning models have been successful in exploiting the relative positioning of data to improve performance. In contrast, the ordering of rows and columns in tabular data don't tend to have meaning, making it a more challenging domain for machine learning models. This challenge is exacerbated by the fact that many collections of tabular data are relatively small [1], meaning they may not be sufficient to effectively train deep models [2].

1.1. Related Works

Despite the difficulties, there have been recent attempts to apply deep learning methods to tabular data. TabNet uses sequential attention to select relevant features, achieving strong performance and interpretable models [3]. FT-Transformer adapted the transformer architecture for tabular applications and reported strong performance across a range of tasks [4]. More recently, TabPFN was introduced, which pretrains a transformer model on synthetically generated datasets yielding a high capacity model which can be adapted to small to medium sized tabular datasets through in-context learning [5].

However, recent benchmarks have found that methods based on decision tree ensembles continue to perform competitively, especially when there is less data available [4][6]. Random forests use an ensemble of small tree predictors created based on a random sampling of features, as the number of trees increases the ensemble's error converges to zero [7]. Even though it was first introduced in 2001, Random forests continue to be a strong baseline of performance across a range of domains [6]. Additionally, there are many variations of algorithms based on gradient boosting [8] which have been found to perform strongly including LightGBM [9], CatBoost [10], and XGBoost [11].

Multitask learning (MTL) is a machine learning approach that looks to improve performance of task performance by learning related tasks in parallel with a common model [12]. There are a variety of MTL techniques that exploit different strategies including: regularization, relationship learning, feature propagation, optimization, and pre-training [13]. MTL has been successfully used in a wide range of domains including dense scene prediction [14], image classification [15], and autonomous driving [16] [17].

There have been some works which bring MTL into the tabular domain. Multi-gate Mixture-of-Experts (MMoE) used a mixture of experts to learn task relationships to improve multitask performance [18]. Progressive Layered Extraction (PLE) then looked to avoid negative transfer between tasks by separating shared and task-specific parts of the network more explicitly [19]. More recently, Shared and Task-specific EMbeddings (STEM) used different embeddings to look to minimize negative transfer [20]. These works differ from the focus of this paper as they are mostly concerned with improving recommendation systems, an application where very large datasets are available whereas this paper focuses on mid-sized datasets.

1.2. Problem Statement

This report investigates the performance of XGBoost [11] on tabular multi-target regression tasks in both single-task and

multi-task settings. Additionally, we compare the impact of standardizing input features and targets on the models' performance. This report focuses on datasets with between 100 and 1000 samples.

2. MULTI-TASK XGBOOST

2.1. XGBoost

XGBoost is a machine learning method built on tree boosting. Gradient tree boosting methods combine regression trees to produce more robust models [8]. In addition to the efficient optimization method described in Section 2.2, XGBoost introduced a novel approach to handling sparse data and added compute-aware optimization to yield a very scalable tree boosting system [11].

Due to its efficiency and strong performance and the release of packages across programming languages [1], XGBoost quickly became a popular tool in machine learning challenges. At the time of its release, the authors reported that 17 of 29 Kaggle challenge winners had used XGBoost in the prior year 2015 [11]. Although deep learning methods have risen in popularity in the intervening decade, XGBoost remains a strong benchmark on tabular machine learning tasks especially when there is not enough data available to sufficiently train more modern deep learning architectures [6].

2.2. Optimization in XGBoost

From the derivation in [11], ensemble tree models usually minimize the regularized objective.

$$\mathcal{L}(\phi) = \sum_i l(\hat{y}_i, y_i) + \sum_k \Omega(f_k) \tag{1}$$
$$\text{where } \Omega(f) = \gamma T + \frac{1}{2}\lambda ||\omega||^2$$

Where l is the differentiable convex loss function and Ω regularizes against model complexity. For each tree f_k, T is the number of leaves and ω are the leaf weights.

Although l is differentiable and convex, because the expression is parameterized by functions (f_k), traditional optimization methods are not available. Instead, gradient boosting optimizes the objective iteratively. At each iteration, a new tree, f_t, is added which greedily minimizes the objective.

$$\mathcal{L}^{(t)} = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)} + f_t(\mathbf{x}_i)) + \Omega(f_t) \tag{2}$$

To iterate more quickly, XGBoost uses the second-order approximation which was originally introduced in [21].

[1] https://xgboost.readthedocs.io/en/stable/index.html

$$\mathcal{L}^{(t)} \simeq \sum_{i=1}^{n} [l(y_i, \hat{y}^{(t-1)}) + g_i f_t(\mathbf{x}_i) + \frac{1}{2} h_i f_t^2(\mathbf{x}_i)] + \Omega(f_t) \tag{3}$$
$$\text{where } g_i = \partial_{\hat{y}^{(t-1)}} l(y_i, \hat{y}^{(t-1)}),\ h_i = \partial^2_{\hat{y}^{(t-1)}} l(y_i, \hat{y}^{(t-1)})$$

This approximated objective function is further simplified by removing the constant terms.

$$\tilde{\mathcal{L}}^{(t)} = \sum_{i=1}^{n} [g_i f_t(\mathbf{x}_i) + \frac{1}{2} h_i f_t^2(\mathbf{x}_i)] + \Omega(f_t) \tag{4}$$

This objective function can be used to evaluate the quality of different tree structures and leaf weights, enabling an exact greedy approach to find the best model. However, this approach can quickly become computationally impractical with larger datasets and continuous features. XGBoost offers a more efficient approximate algorithm which uses candidate split points to essentially discretize continuous features, further improving the method's efficiency [11].

2.3. Multi-output Regression in XGBoost

In the original work, XGBoost was tested on classification, ranking, and regression tasks but there were no experiments with a multi-output setup [11]. By default, the XGBoost package handles multiple outputs with a separate model for each output without information sharing between the tasks.

However, the package can be adapted to a native multi-output setup where trees output a vector of the multiple outputs rather than a single value. The optimization and objective function presented in Section 2.2 can be trivially extended to handle vectors as predictions. In equation 3, y_i and $\hat{y}$ become vectors $\mathbf{y}_i$ and $\hat{\mathbf{y}}$ respectively and the loss function l is extended accordingly. In this case each addition of a tree no longer greedily optimizes for a single task, rather it improves the average loss across all the outputs.

3. EXPERIMENTAL DETAILS

3.1. Datasets

Experiments used three multi-target regression datasets from the benchmark introduced in [22]: Jura heavy metal contamination dataset (Jura), Energy building dataset (ENB), and Airline ticket price dataset (ATP1D). These datasets were selected because they have between 100 and 1000 samples, an amount of data where gradient boosting methods like XGBoost have remained near the state of the art [6]. More details on the dataset characteristics are available in Table 1.

Dataset	# Samples	# Features	# Targets
Jura [23]	359	15	3
ENB [24]	768	8	2
ATP1D [25]	337	370	6

Table 1. Dataset characteristics

3.2. Standardization

In trials with standardization, the input features and targets have undergone z-score standardization. Every input feature and target value, x has been transformed to x' with a mean of 0 and standard deviation of 1.

$$x' = \frac{x - \mu_x}{\sigma} \tag{5}$$

Standardization could improve the performance of resulting models because it evens out the magnitudes of features, approximately reducing the anisotropy of the feature and target space which can make gradient descent methods more efficient [26]. Additionally, in a multi-task setup this could help even out the different tasks contribution to the combined loss.

3.3. Hyperparameter Search

For all trials, we conduct a hyperparameter search and report the performance of the best model. In all cases, we vary the number of estimators, the max tree depth, and the learning rate; conducting a grid search across all 72 combinations of the parameters in Table 2.

Parameter	Search Space
Number of estimators	10, 50, 100, 500, 1000, 3000
Max tree depth	1, 2, 3, 5
Learning rate	0.1, 0.01, 0.001

Table 2. Hyperparameter search space

For each hyperparameter configuration, we evaluate the model with 5-fold cross-validation, averaging the R^2 values across the five folds. For the single-task models, it is trivial to then select the model with the highest average R^2. For the multi-task models, each model has an average R^2 value for each task so the values are combined across tasks with a clipped L2 norm. Clipping is necessary so that extreme negative R^2 values don't skew the comparison. The clipped norm is defined as:

$$||\tilde{R}^2||_2 = ||\max(R^2, 0)||_2 = \sqrt{\sum_{i=1}^{n} \max(R_i^2, 0)} \tag{6}$$

There could be a risk that this approach would allow a model to disregard one task since the worst performance for any task is limited to 0. However it is observed that none of the models take advantage of this.

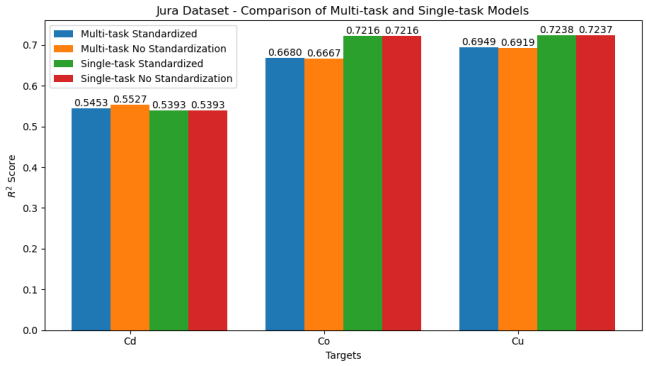

Fig. 1. Results of XGBoost models on Jura Dataset.

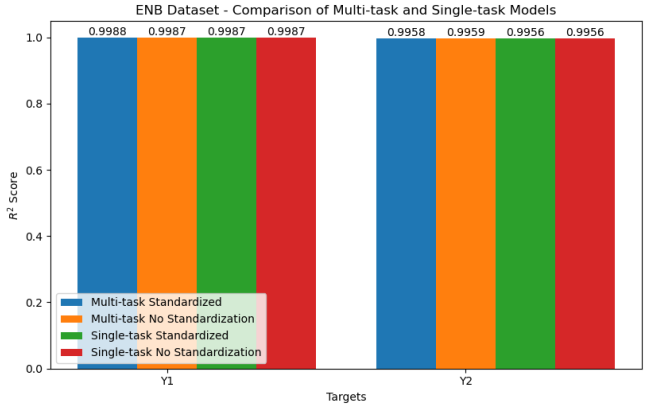

Fig. 2. Results of XGBoost models on ENB Dataset.

4. RESULTS & ANALYSIS

4.1. Single-task and multi-task models

In Figure 1, we see that the single-task models outperform the multi-task models on 2 of 3 tasks. For Co, the multi-task models have an R^2 around 0.67 and the single-task models have an R^2 around 0.72. For Cu, the multitask models have an R^2 around 0.72 and the single-task models have an R^2 around 0.66. On the remaining task, Cd, the performance between model types is very close with R^2 scores close to 0.55 and 0.54 for the multi-task and single-task models respectively.

In Figure 2, we see that the performance across all model types is saturated near an R^2 of 1. This makes it difficult to distinguish any benefits of either model type.

Figure 3 shows mixed results, in 4 of 6 tasks the multi-task models outperform the single-task models but the opposite is true in the remaining tasks. The magnitude of difference in performance between model types is less than 3% in all cases.

Across the three datasets, we observe mixed results in comparing multi-task and single-task applications of XGBoost. With the Jura dataset, the single-task models outperform on 2 of 3 tasks; with the ENB dataset, there is no

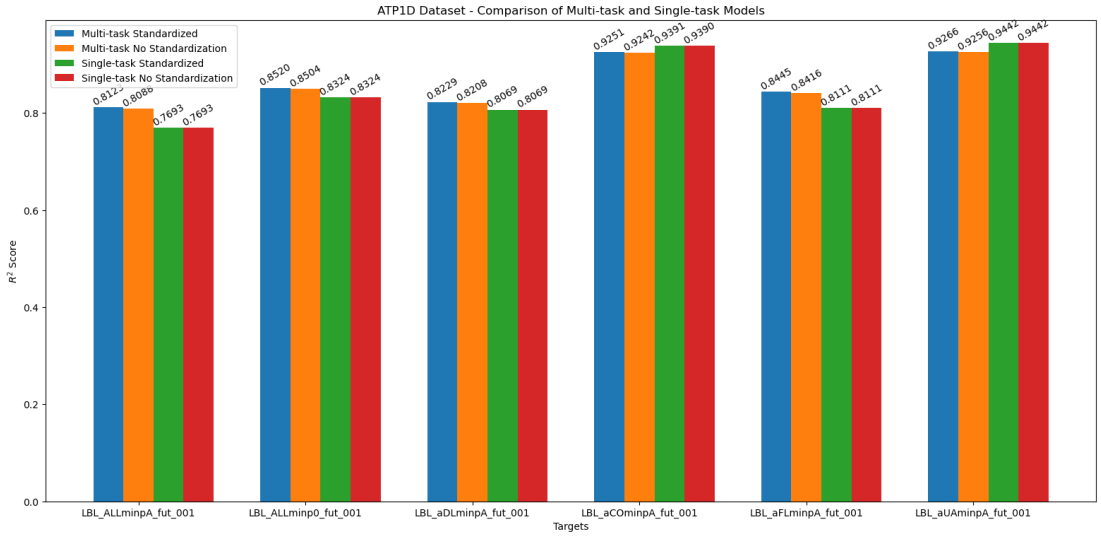

Fig. 3. Results of XGBoost models on ATP1D Dataset.

meaningful difference between multi-task and single-task performance; with the ATP1D dataset, the multi-task strategy improves performance on 4 of 6 tasks. From these results, we can conclude that multi-task XGBoost may be an approach worth exploring in some circumstances but should not be applied blindly in place of single-task models.

4.2. Effect of standardization

Across Figures 1, 2, and 3 we can see that there is no meaningful difference between the models trained with standardized and non-standardized data. This suggests that the differences in magnitudes of features and targets were not impeding the performance of XGBoost for these datasets. Given the lack of benefit in terms of accuracy, the non-standardized models would be preferred since they require less data preprocessing.

5. CONCLUSION

In this work we find that single-task and multi-task applications of XGBoost outperform each other in certain circumstances with different datasets. Given a new dataset and problem space, one should explore both options to determine which approach is best for that specific application.

This report found inconclusive results with XGBoost but there could be value in evaluating the potential of integrating MTL into other prominent gradient boosting methods. It is also possible that the tree ensemble models don't have enough capacity to learn a robust shared representation. This might motivate further work on applying deep models to small and mid-sized models. Future work integrating pre-training with synthetic data like [5] into a multi-task framework would be another promising direction for future work.

6. REFERENCES

[1] Michael Chui, James Manyika, Mehdi Miremadi, Nicolaus Henke, Rita Chung, Pieter Nel, and Sankalp Malhotra, "Sizing the potential value of AI and advanced analytics," Tech. Rep., McKinsey Global Institute, Apr. 2018.

[2] Ian Goodfellow, Yoshua Bengio, and Aaron Courville, *Deep Learning*, MIT Press, 2016.

[3] Sercan Ö Arik and Tomas Pfister, "TabNet: Attentive Interpretable Tabular Learning," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 8, pp. 6679–6687, May 2021, Number: 8.

[4] Yury Gorishniy, Ivan Rubachev, Valentin Khrulkov, and Artem Babenko, "Revisiting Deep Learning Models for Tabular Data," in *Advances in Neural Information Processing Systems*. 2021, vol. 34, pp. 18932–18943, Curran Associates, Inc.

[5] Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter, "Accurate predictions on small data with a tabular foundation model," *Nature*, vol. 637, no. 8045, pp. 319–326, Jan. 2025, Publisher: Nature Publishing Group.

[6] Assaf Shmuel, Oren Glickman, and Teddy Lazebnik, "A Comprehensive Benchmark of Machine and Deep Learning Across Diverse Tabular Datasets," Aug. 2024, arXiv:2408.14817 [cs].

[7] Leo Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, Oct. 2001.

[8] Jerome H. Friedman, "Greedy function approximation: A gradient boosting machine.," *The Annals of Statistics*, vol. 29, no. 5, pp. 1189–1232, Oct. 2001, Publisher: Institute of Mathematical Statistics.

[9] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu, "LightGBM: A Highly Efficient Gradient Boosting Decision Tree," in *Advances in Neural Information Processing Systems*. 2017, vol. 30, Curran Associates, Inc.

[10] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin, "CatBoost: unbiased boosting with categorical features," in *Advances in Neural Information Processing Systems*. 2018, vol. 31, Curran Associates, Inc.

[11] Tianqi Chen and Carlos Guestrin, "XGBoost: A Scalable Tree Boosting System," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, New York, NY, USA, 2016, KDD '16, pp. 785–794, Association for Computing Machinery.

[12] Rich Caruana, "Multitask Learning," *Machine Learning*, vol. 28, no. 1, pp. 41–75, July 1997.

[13] Jun Yu, Yutong Dai, Xiaokang Liu, Jin Huang, Yishan Shen, Ke Zhang, Rong Zhou, Eashan Adhikarla, Wenxuan Ye, Yixin Liu, Zhaoming Kong, Kai Zhang, Yilong Yin, Vinod Namboodiri, Brian D. Davison, Jason H. Moore, and Yong Chen, "Unleashing the Power of Multi-Task Learning: A Comprehensive Survey Spanning Traditional, Deep, and Pretrained Foundation Model Eras," Apr. 2024, arXiv:2404.18961 [cs].

[14] Dimitrios Sinodinos and Narges Armanfard, "EMA-Net: Efficient Multitask Affinity Learning for Dense Scene Predictions," Jan. 2024, arXiv:2401.11124 [cs].

[15] Shikun Liu, Edward Johns, and Andrew J. Davison, "End-to-End Multi-Task Learning with Attention," Apr. 2019, arXiv:1803.10704.

[16] Zhongyu Xia, ZhiWei Lin, Xinhao Wang, Yongtao Wang, Yun Xing, Shengxiang Qi, Nan Dong, and Ming-Hsuan Yang, "HENet: Hybrid Encoding for End-to-end Multi-task 3D Perception from Multi-view Cameras," Apr. 2024.

[17] Jyh-Jing Hwang, Runsheng Xu, Hubert Lin, Wei-Chih Hung, Jingwei Ji, Kristy Choi, Di Huang, Tong He, Paul Covington, Benjamin Sapp, Yin Zhou, James Guo, Dragomir Anguelov, and Mingxing Tan, "EMMA: End-to-End Multimodal Model for Autonomous Driving," Nov. 2024, arXiv:2410.23262 [cs] version: 2.

[18] Jiaqi Ma, Zhe Zhao, Xinyang Yi, Jilin Chen, Lichan Hong, and Ed H. Chi, "Modeling Task Relationships in Multi-task Learning with Multi-gate Mixture-of-Experts," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, New York, NY, USA, 2018, KDD '18, pp. 1930–1939, Association for Computing Machinery, event-place: London, United Kingdom.

[19] Hongyan Tang, Junning Liu, Ming Zhao, and Xudong Gong, "Progressive Layered Extraction (PLE): A Novel Multi-Task Learning (MTL) Model for Personalized Recommendations," in *Proceedings of the 14th ACM Conference on Recommender Systems*, New York, NY, USA, 2020, RecSys '20, pp. 269–278, Association for Computing Machinery, event-place: Virtual Event, Brazil.

[20] Liangcai Su, Junwei Pan, Ximei Wang, Xi Xiao, Shijie Quan, Xihua Chen, and Jie Jiang, "STEM: Unleashing the Power of Embeddings for Multi-Task Recommendation," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 8, pp. 9002–9010, Mar. 2024, Number: 8.

[21] Jerome Friedman, Trevor Hastie, and Robert Tibshirani, "Additive logistic regression: a statistical view of boosting (With discussion and a rejoinder by the authors)," *The Annals of Statistics*, vol. 28, no. 2, pp. 337–407, Apr. 2000, Publisher: Institute of Mathematical Statistics.

[22] Eleftherios Spyromitros-Xioufis, Grigorios Tsoumakas, William Groves, and Ioannis Vlahavas, "Multi-target regression via input space expansion: treating targets as inputs," *Machine Learning*, vol. 104, no. 1, pp. 55–98, July 2016.

[23] Joao Negreiros and Ana Neves, "Spatial Analysis with myGeoffice.org: The Pb Jura Lake, Switzerland, Contamination Case," *Journal of Geoscience and Environment Protection*, vol. 7, no. 2, pp. 92–113, Feb. 2019, Number: 2 Publisher: Scientific Research Publishing.

[24] Athanasios Tsanas and Angeliki Xifara, "Accurate quantitative estimation of energy performance of residential buildings using statistical machine learning tools," *Energy and Buildings*, vol. 49, pp. 560–567, June 2012.

[25] William Groves and Maria Gini, "On Optimizing Airline Ticket Purchase Timing," *ACM Trans. Intell. Syst. Technol.*, vol. 7, no. 1, pp. 3:1–3:28, Oct. 2015.

[26] Stephen Boyd and Lieven Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004.