Machine Learning Application for Smart Network Traffic Prediction

Islam Omar[1], Whit Schonbein[2], Abdel-Hameed A. Badawy[1]
[1]Klipsch School of Electrical and Computer Engineering, New Mexico State University, Las Cruces, NM, USA
[2]Center for Computing Research, Sandia National Laboratories, Albuquerque, NM, USA
{islam, badawy}@nmsu.edu, wwschon@sandia.gov

***Abstract*—With the ever-growing usage of the internet and the larger loads being generated in recent years, the prediction of network traffic has become critical for efficient network management and optimization. In this work, we take an initial step toward a comprehensive study of the application of machine learning models for network traffic prediction. Using real-world traffic data collected at Sandia National Labs for several scientific applications, we investigate the features of these datasets and how different preprocessing methods affect the behavior of different machine learning models. Specifically, we evaluate the performance of several machine learning models, including Linear Regression (LR), Support Vector Machine (SVM), Random Forest (RF), and XGBoostRegressor (XGBR), against the AutoRegressive Integrated Moving Average (ARIMA) model. The models are trained and tested on both original and normalized versions of the network traffic dataset to understand the impact of data normalization on predictive accuracy and how the performance of the machine learning models varies based on the nature of the problem and the provided data.**

***Index Terms*—Network Traffic, Machine Learning, Forecasting**

I. Introduction

In this paper, we are trying to highlight the importance of applying machine learning algorithms for forecasting the load on a network based on the nature of the usage of that specific program from taking this information for several runs of the same program. Another observation we highly recommend is paying attention to the used machine learning models and the evaluation metrics that would give misleading evaluation numbers that don't represent the performance of these models. We have built several machine learning models from different categories to have a diversity in the used models to make sure that one of them would converge with our dataset after making them more training-friendly for our machine learning models using several preprocessing techniques such as normalization, undersampling, and principle component analysis (PCA) to come up with the best combination of features for our machine learning models.

II. Methodology

This study conducts a comprehensive evaluation of various machine learning models and compares their performance against the ARIMA model and the impact of normalizing the training and testing datasets using the StandardScaler (SC) normalization technique. The machine learning models applied are Linear Regression (LR), Decision Linear Regression (DLR), Random Forest (RF), XGBoostRegressor (XGBR), Support Vector Machine (SVM), and Long Short-Term Memory (LSTM) model which is the AutoRegressive Integrated Moving Average (ARIMA) model.

Using real-world traffic data collected at Sandia National Labs for several scientific applications, Our datasets consist of six runs for each of nine scientific applications collected from real-life traffic data at Sandia National Labs. The data is big enough from a vertical perspective, with a number of instances in each run, but the challenge comes in two perspectives; the first one is the horizontal perspective of the datasets, the number of features. The second challenge is the shape of the trend that comes out from the visualization of some of the applications as shown in Figure 1.

We have considered the AutoRegressive Integrated Moving Average (ARIMA) model before any other types of machine learning models especially since we don't have the sufficient number of features that most other machine learning models would require to converge with our datasets. We still considered classic models such as Classic Linear Regression (CLR) -to be our base model-, Support Vector Machine (SVM), and decision tree-based models such as RandomForest (RF) and XGBoostRegressor (XGBR) which have proven to be able to handle the lack of training features and would rely on the trend on the target column -network traffic load in our case-. We have tried several approaches to splitting the datasets into training and testing sub-datasets. First, we tried to train the models on five runs and test them against the sixth one which

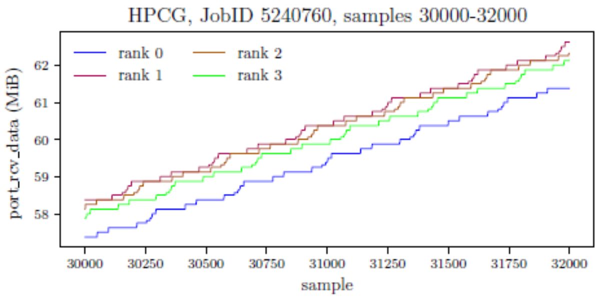
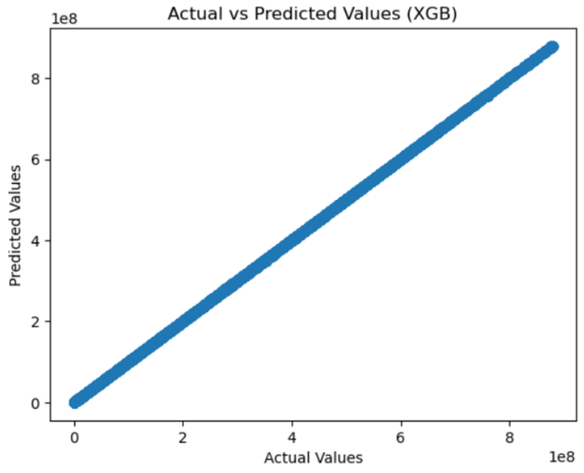


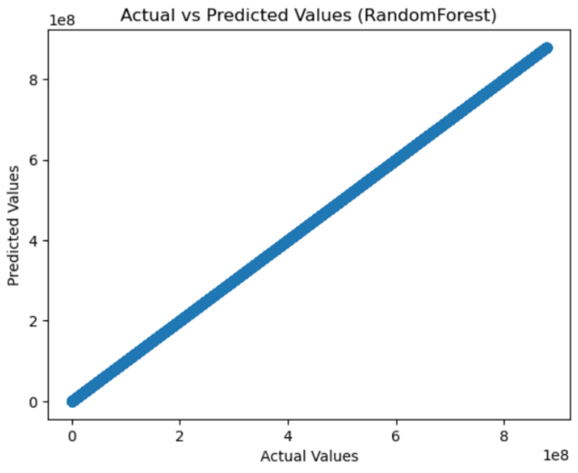
Fig. 1: Visualization for our HPCG program with different ranks showing how the trend changes from one rank to another and the non-normal behavior of the trend visualizations

```
MSE: 627524238509.8503
RMSE: 792164.2749517617
R-squared Score: 0.9999856040866342
```



(a) The visualization for the actual testing values sorted on the Y-Axis and their corresponding predicted values produced using XGBoost Regression model plotted on the X-Axis with the evaluation metrics MSE, RMSE, and R2-Score values

```
MSE: 639374011.6687043
RMSE: 25285.846073815766
R-squared Score: 0.9999999853322432
```



(b) The visualization for the actual testing values sorted on the Y-Axis and their corresponding predicted values produced using RandomForest Regression model plotted on the X-Axis with the evaluation metrics MSE, RMSE, and R2-Score values

Fig. 2: These graphs represent another validation approach we took to make sure our XGBR and RF models are giving good results, and we can compare these results with the evaluation metrics values to see which metric was evaluating the performance of the models in a more accurately way based on the actual results we are getting from producing values using them and comparing these values to the actual testing values

is the closest to the 80-20 stereotype training-testing split and we tried to combine all the runs for each program into one big dataset and split it afterward into 80% training and 20% testing datasets which gave the same results disclosed in the results section, however, with way more training time and required computational power due to the size of the datasets. Based on that we decided to check how minimizing the training dataset would affect the results of the models based on the evaluation metrics and that was our third and final approach that gave the results shared in the results sections. For building CLR, SVM, RF, and XGBR models, we used GirdSearchCV() to find the best combination of values for their hyperparameters and make sure our models are not overfitting the training dataset. For evaluation of the performance of all of our machine learning models, classic ones and the ARIMA, we have used four evaluation metrics; Mean Absolute Error (MAE), Mean Square Error (MSE), Root Mean Squared Error (RMSE), and R2-Score (R2) to have diversity in our evaluation metrics.

III. Results

the ARIMA model showed some indications that it would overcome the rank feature effect on the trend of the programs' datasets from the specs we came up with, however, the MAE and MSE values were too high and the R2-Score were way too low indicating that the model didn't coverage well with the training dataset causing the ARIMA model to overfit the training dataset and giving very bad results when tested against our testing sub-dataset. Including more runs -More instances- in the training dataset, but still, the ARIMA model's performance didn't improve emphasizing that the amount of data provided in the training stage, after a specific point, does not add any new information for the model. The RF and XGBR models showed similar very high MAE and MSE values, however, they showed at the same time a 99% R2-Score value which required an additional step of validation for our models' performance to see which evaluation metric is more reliable in such case as ours. We have excluded the CLR, and SVM models' results as they didn't converge with the training datasets. Visualization has shown a solid linear relation between the actual testing values and the predicted ones by the RF and XGBR models. The comparison visualization illustrated in Figure 2b and Figure 2a emphasizes that the R2-Score is more reliable in a research case as our forecasting issue with this format of data as the MAE and MSE evaluation metrics' equations require the predicted values to be identical to actual ones which can be tolerated in our case as long as the models did converge and give very acceptable forecasted values.

IV. Conclusion

We conclude that machine learning models are to be considered for network traffic load forecasting and how they can converge and be reliable even with lacking enough features taking into consideration the size of the data being provided and the preprocessing techniques that should be considered before passing the data into its original format to the models for making any decisions further on. The ARIMA model was over-performed by decision tree-based models such as RF and XGBR models that showed high flexibility even with the lacking features and were able to handle the effect of the rank feature on the trend of our target column, and network traffic load, even when we minimized the training data.

Normalization is not always going to provide additional information for the models or guarantee any better performance from them, we would like to emphasize the same hypothesize as we saw that using StandardScaler() as our normalization function did not improve the performance of our machine learning models based on the nature of our datasets that already following a normal trend. More data does not always mean that the models will perform better as it sometimes does not provide any new information for the models and at the same time costs the machine learning engineering more computational power and Time. Evaluation metrics are to be overlooked carefully based on the results of the model and the nature of the problem we are trying to solve as some of the evaluation metrics such as MAE and MSE do not have enough tolerance to see that the performance of the models is providing the results we are looking for, therefore, we recommend always to check and compare the forecasting result of the models and manually compare them against the actual testing values.

References

[1] W. McKinney, *Data Structures for Statistical Computing in Python*, in *Proceedings of the 9th Python in Science Conference*, 2010, pp. 51-56. The ‘pandas‘ library is introduced by Wes McKinney, and this paper is the foundational reference.

[2] S. Seabold and J. Perktold, *statsmodels: Econometric and Statistical Modeling with Python*, in *Proceedings of the 9th Python in Science Conference*, 2010, pp. 92-96. The ‘statsmodels‘ library provides tools for statistical modeling and econometrics in Python, as described in this paper.

[3] G. E. P. Box, G. M. Jenkins, and G. C. Reinsel, *Time Series Analysis: Forecasting and Control*, 3rd ed., Prentice Hall, 1994. ARIMA models are discussed extensively in this book.

[4] M. S. Kendall and A. Stuart, *The Advanced Theory of Statistics*, Vol. 2: Inference and Relationship, 4th ed., Charles Griffin & Company Ltd., 1979. Partial correlation is discussed as part of the statistical theory and analysis of relationships.

[5] G. E. P. Box, G. M. Jenkins, and G. C. Reinsel, *Time Series Analysis: Forecasting and Control*, 3rd ed., Prentice Hall, 1994. Autocorrelation is a key concept in time series analysis, discussed extensively in this book.

[6] J. D. Hamilton, *Time Series Analysis*, Princeton University Press, 1994. Data differencing is a technique used to make a time series stationary, covered in this comprehensive text on time series analysis.

[7] W. W. S. Wei, *Time Series Analysis: Univariate and Multivariate Methods*, 2nd ed., Addison-Wesley, 2006. The concept of stationarity in time series data is fundamental and is covered in detail in this book.

[8] C. Cortes and V. Vapnik, *Support-Vector Networks*, Machine Learning, vol. 20, no. 3, pp. 273-297, 1995. This paper introduces the concept of Support Vector Machines.

[9] A. C. Rencher, *Linear Models in Statistics*, 2nd ed., John Wiley & Sons, 2008. Classic Linear Regression is thoroughly covered in this book.

[10] L. Breiman, *Random Forests*, Machine Learning, vol. 45, no. 1, pp. 5-32, 2001. This paper provides the foundational details of the Random Forest algorithm.

[11] T. Chen and C. Guestrin, *XGBoost: A Scalable Tree Boosting System*, in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD ’16)*, 2016, pp. 785-794. This paper presents the XGBoost algorithm and its applications.

[12] I. T. Jolliffe, *Principal Component Analysis*, Springer Series in Statistics, 2nd ed., Springer-Verlag, 2002. The concept of feature scaling, including standardization, is discussed in the context of data preprocessing.

[13] R. A. Johnson and D. W. Wichern, *Applied Multivariate Statistical Analysis*, 6th ed., Pearson, 2007. Polynomial feature transformation is part of polynomial regression, covered in this book.

[14] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen, *Classification and Regression Trees*, Wadsworth International Group, 1984. Feature contribution, or importance, is often discussed in the context of decision trees and ensemble methods.

[15] D. J. Hand and C. C. Taylor, *Multivariate Analysis of Variance and Repeated Measures: A Practical Approach for Behavioural Scientists*, Chapman and Hall/CRC, 1987. Mean Absolute Error (MAE) is a common regression evaluation metric discussed in various statistical and machine learning texts.

[16] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, Springer Series in Statistics, 2nd ed., Springer, 2009. Mean Squared Error (MSE) is a widely used error metric in regression analysis, covered extensively in this book.

[17] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, Springer Series in Statistics, 2nd ed., Springer, 2009.

[18] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning: with Applications in R*, Springer, 2013.