# 2. Related Work

Machine learning algorithms have become increasingly popular in recent years as a means of accurately predicting performance or behavior in various fields. Several previous works have attempted to develop models for this purpose. For instance, Fu et al. [345] applied machine learning to predict the performance of applications and use-cases, but the prediction error remained high for several realistic scenarios. Some research [?] applied machine learning algorithms to predict computer operating system security using dataset from program and determined whether a newly discovered operating system vulnerability would enable attackers to cause denial of service to the subject system by predictive models. Akgun st al. [?] thought of building a light-weight machine learning engine in kernel space component, which is a progressive idea to combine kernel space and machine learning together since machine learning usually happens on user space.

The topic of process performance has been widely studied in the field of computer science. Aslam et al. [123] focused on improving CPU scheduling by applying Bayesian Decision Theory. Ardalani et al. [56] proposed a machine-learning-based tool, Cross-Architecture Performance Prediction (XAPP), that utilized a single-threaded CPU implementation to predict GPU performance. XAPP leverages established machine learning techniques to learn the correlation between program properties, hardware characteristics, and GPU execution time, achieving 26.9% average error on a set of 24 real-world kernels and providing a highly desirable tool for estimating GPU performance before writing a GPU implementation.

While many studies have focused on predicting process performance in specific contexts, there is a need for more general models that can accurately predict process performance across a wide range of scenarios. The present study aims to address this gap by developing predictive machine learning models that can accurately estimate the performance of processes on both Linux and macOS operating systems, using data collected from a diverse range of users.

## 4.1.3 Predictive Models

In this study, the Python library sklearn was utilized to construct various machine learning models, including linear regression and random tree regression. To evaluate the performance of each model, various metrics were applied including Mean Squared Error, R2 score, and cross-validation scores. For cross validation, the function cross\_val\_score was used to estimate model's accuracy in a more robust way than with just the typical train-test split. All the data were fed inside it, and the function made the necessary train-test splits, compared with using train\_test\_split.

Part of model validation for linear regression model was applied. The linear relationship was checked by scatter plotting Actual value Vs Predicted value and multivariate normality was best check with Q-Q plot. Also, residual Vs fitted value scatter plot was checked to make sure heteroscedastic plot exhibited a funnel shape pattern.

The random tree regression model was specifically set with 1000 decision trees running in the model and 42 random states, and several parameters were adjusted to determine the criterion used to determine model outcomes, the maximum possible depth of each tree, and the maximum number of features the model considered when determining a split.

The model's consistency was further evaluated using Mean Squared Error, R2 score, and cross-validation scores. During cross-validation, RepeatedKFolder function from sklearn was utilized to determine the number of splits to use. This cross-validator repeats the K-Fold n times with different randomization in each repetition. The robustness of results was ensured by splitting the data into 5 folds and repeating the process 10 times, each with different random splits. This configuration could evaluate the performance of machine learning models on a limited sample of data in a more robust manner.

## 4.2. Experiment Result

Apple

RFR

MSE(9.381556e+08 on macOS data and ??? on Linux data)

RMSE(30629.325091 on macOS data and ???)

R2(0.998830 on macOS data and 0.999998 on Linux data)

LR

(3.895207e+11 on macOS data and ??? on Linux data)

(624115.918959 on macOS data and ???? on Linux data)

(0.514342 on macOS data and 0.173071 on Linux data)

Apple

Method Test MSE Test RMSE Test R2 Cross-Val MSE Mean Cross-Val RMSE Mean Cross-Val MSE Standard Deviation

0 Random forest 6.636878e+10 257621.382557 0.992468 9.660352e+10 302881.154619 6.065075e+10

linux

Method Test MSE Test RMSE Test R2 Cross-Val MSE Mean Cross-Val RMSE Mean Cross-Val MSE Standard Deviation

0 Linear regression 8.805821e+15 9.383933e+07 0.173071 7.684483e+15 8.733552e+07 1.325742e+15

1 Random forest 2.372822e+10 1.540397e+05 0.999998 1.340514e+11 2.196216e+05 3.277659e+11

# 7. Reference

[7] Montgomery D. C., Peck E. A. and Vining G. C. Introduction to Linear Regression Analysis. Wiley, New York, NY, 2001.

[123][N. Aslam, N. Sarwar, A. Batool. Designing a Model for improving CPU Scheduling by using Machine Learning. In Proc. of (IJCSIS) International Journal of Computer Science and Information Security, Vol. 14, No. 10, Oct 2016](https://d1wqtxts1xzle7.cloudfront.net/51178809/25_Paper_30091654_IJCSIS_Camera_Ready_pp._201-204-libre.pdf?1483532507=&response-content-disposition=inline%3B+filename%3DDesigning_a_Model_for_Improving_CPU_Sche.pdf&Expires=1681967949&Signature=K1B8MJz86KVRkUnkCZ3Ex2Bu~IiK~wUZKw7K4umy6OfykU~pHPlmOKnWe5zRDxRp6V8BcOo3jBRpBzm5svb-uTE4qpO2kyvv91Qn2zfRAFvGQBW~gpxPyCMTrVywl40GjQOrSTNB3kEKm5ixK-PE9x6~Ur5Q8UwM159D9oCLJrYCnkaVDxg3Ee7pVny958vtUZpIuqCyc~AGi6~8WOusZRnLhwBmwNve8fs7GMtpwzqZTIe~0JYbYhM4bOkcdMGbwlMIUmHlmkV7ho6UUhNKFZn0XDRkIpabKPa-ZGyE-unyKJxxPWscII8Mo-lcg7VvoBqPQ9GHG4g92n3FtV5Agg__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA)

[345]S. Fu, S. Gupta, R. Mittal, S. Ratnasamy. On the Use of ML for Blackbox System Performance Prediction. In Proc. of the 18th USENIX Symposium on Networked Systems Design and Implementation, Apr 2021

[234] [Hiba Asri a, Hajar Mousannif b, Hassan Al Moatassime c, Thomas Noel d. Using Machine Learning Algorithms for Breast Cancer Risk Prediction and Diagnosis. The 6th International Symposium on Frontiers in Ambient and Mobile Systems. 12 May 2016.](https://www.sciencedirect.com/science/article/pii/S1877050916302575)

[456] [Jianbo Liu a, Dragan Djurdjanovic a, Jun Ni a, Nicolas Casoetto a, Jay Lee b. Similarity based method for manufacturing process performance prediction and diagnosis. Computers in Industry. Volume 58, Issue 6, August 2007, Pages 558-566. December 2006](https://www.sciencedirect.com/science/article/abs/pii/S0166361506001850)

[56] [Newsha Ardalani, Clint Lestourgeon, Karthikeyan Sankaralingam, Xiaojin Zhu. Cross-architecture performance prediction (XAPP) using CPU code to predict GPU performance. MICRO-48: Proceedings of the 48th International Symposium on MicroarchitectureDecember](https://dl.acm.org/doi/abs/10.1145/2830772.2830780)

[[] Machine Learning Approach to Predict Computer Operating Systems Vulnerabilities](https://ieeexplore.ieee.org/document/9096731/authors" \l "authors)

[Freeh Alenezi; Chris P. Tsokos](https://ieeexplore.ieee.org/document/9096731/authors" \l "authors)

[Published in: 2020 3rd International Conference on Computer Applications & Information Security (ICCAIS)](https://ieeexplore.ieee.org/document/9096731/authors" \l "authors)

[Date of Conference: 19-21 March 2020](https://ieeexplore.ieee.org/document/9096731/authors" \l "authors)

[Date Added to IEEE Xplore: 20 May 2020](https://ieeexplore.ieee.org/document/9096731/authors" \l "authors)

[[?]KMLIB: TOWARDS MACHINE LEARNING FOR OPERATING SYSTEMS](https://www.fsl.cs.stonybrook.edu/~umit/files/kmllib-sysml-paper.pdf)

[Ibrahim Umit Akgun 1 Ali Selman Aydin 1 Erez Zadok 1](https://www.fsl.cs.stonybrook.edu/~umit/files/kmllib-sysml-paper.pdf)

[?]A. Maros, F. Murai, A. P. Couto da Silva, J. M. Almeida, M. Lattuada, E. Gianniti, M. Hosseini, and D. Ardagna. “Machine learning for performance prediction of spark cloud applications,” In 2019 IEEE 12th International Conference on Cloud Computing (CLOUD), pages 99–106. IEEE, July 2019

by M. Abdollahzadeh, M. K. Akbari, and M. Sadeghizadeh, published in the International Journal of Computer Science and Information Security, Volume 14, Issue 1, January 2016, Pages 201-204.

[?]https://www.usenix.org/system/files/osdi18-lee.pdf

[?]https://ieeexplore.ieee.org/document/4085157

[?]https://kodu.ut.ee/~dumas/pubs/WhiteBoxJSEP.pdf

[?]http://alumni.cs.ucr.edu/~kishore/papers/tencon.pdf

[?]https://www.researchgate.net/publication/220938947\_Efficiently\_exploring\_architectural\_design\_spaces\_via\_predictive\_modeling

[?]https://dl.acm.org/doi/abs/10.1145/325164.325119

[?]https://www.seas.upenn.edu/~leebcc/documents/lee2006-asplos.pdf

https://dev.to/arepp23/how-to-write-to-a-csv-file-in-c-1l5b

https://towardsdatascience.com/how-to-build-your-first-machine-learning-model-in-python-e70fd1907cdd

https://towardsdatascience.com/how-to-use-random-seeds-effectively-54a4cd855a79

https://www.analyticsvidhya.com/blog/2021/05/know-the-best-evaluation-metrics-for-your-regression-model/

https://www.linuxjournal.com/article/8110

# Topic:

performance. I can also use classification algorithms such as decision trees or neural networks to classify the performance as good or bad based on the input data. So sounds straightforward.

3. Predictive Models

In addition, I have also encountered some challenges for this project.

Thus it is Not efficient enough to collect linux data in this way. Compared with the data collecting approach I applied on macbook, it still has some advantage that the data is recorded constantly on virtual file. I can still choose to export large gigabytes of data without missing consistent data info since the kernel module is working.

~~In our experimentation we modify the Linux Kernel scheduler to allow scheduling with customized time slices. The "Waikato Environment for Knowledge Analysis" (Weka), an open source machine-learning tool is used to find the most suitable ML method to characterize our programs. We experimentally fined that the C4.5 Decision Tree algorithm most effectively solved the problem. We fined that predictive scheduling could reduce TaT in the range of 1.4% to 5.8%. This was due to a reduction in the number of context switches needed to complete the process execution. We find our result interesting in the context that generally operating systems presently never make use of a program's previous execution history in their scheduling behavior.~~