Dicle University
Journal of Engineering
https://dergipark.org.tr/tr/pub/dumf
duje.dicle.edu.tr

Research Article

Predicting CPU Performance Score with Regression Analysis

Güney Kaya[1*], İzzet Emre Şen[2], Osman Altay[3]

[1] Manisa Celal Bayar University, Department of Software Engineering, guneykaya2002@gmail.com ORCID: 0009-0007-7687-7350
[2] Manisa Celal Bayar University, Department of Software Engineering, emre.senn83@gmail.com ORCID: 0009-0006-8640-8374
[3] Manisa Celal Bayar University, Department of Software Engineering, osman.altay@cbu.edu.tr ORCID: 0000-0003-3989-2432

ARTICLE INFO

Article history:

Received 31 May 2024
Received in revised form 8 January 2025
Accepted 10 January 2025
Available online 26 March 2025

Keywords:

Regression Analysis, Machine Learning, Data Mining, CPU, CPU Performance

Doi: 10.24012/dumf.1493049

* Corresponding author

ABSTRACT

A central processing unit (CPU) is the core component of a computer that processes data and executes commands, directly impacting the system's performance. Choosing the right processor is crucial, as it ensures efficient, rapid operation and supports the overall system's functionality. An optimal processor choice significantly influences device performance, making it a critical factor in system design. In this research, the CPU's performance score was predicted using regression analysis, based on its features. Support Vector Regression (SVR), Random Forest Regression (RFR), Multiple Linear Regression (MLR), Gradient Boosting Regression (GBR) and Neural Network Regression (NNR) are used to estimate the CPU's performance score. As a result, the NNR has the highest coefficient of determination score, which is 0.9765, followed by GBR, 0.9588. MLR, RFR, and SVR algorithms have the R-squared score of 0.9522, 0.9346, and 0.8658, respectively.

Introduction

In our modern world, computers are so widely used that evaluating the performance of a computer system, or more specifically, the central processing unit (CPU) of a computer, is incredibly important in various decisions [1]. A computer's CPU performs various functions such as calculations, logical decisions, transferring data from one place in the computer's memory to another, and multitasking between applications running on your desktop. Therefore, CPU performance is not only incredibly important for choosing your next computer, but also for computer system configuration and system design.

Research has explored the use of performance counters and power weights to create a prediction model for CPU power consumption and potential applications in power-aware embedded systems, mapping hardware performance counter values to processor power consumption and utilizing this model to predict runtime and CPU power consumption [2]. Machine learning regression algorithms are increasingly used to predict CPU performance based on hardware components and other observable data. Regression analysis enables indirect prediction of resource consumption and performance based on easily observed data [3]. For example, one study proposed a number of methods to predict CPU performance based on hardware characteristics such as the amount of memory and processor speed [4].

In our case, we decided that it would be appropriate for us to use regression analysis to understand how CPU performance scores vary with CPU specifications. Regression analysis is a statistical method that helps understand how one variable changes with respect to another variable. This research paper aims to predict CPU performance scores from CPU features using regression analysis. To do so, a dataset containing various CPU features will be used to create a regression model. The accuracy and reliability of the model will then be evaluated using test data. This research will help to predict CPU performance scores more accurately, providing computer users with better purchasing options. Users will be able to avoid wasting money and time by selecting a CPU that fits their desired performance level.

Additionally, this research might benefit the CPU production process and contribute to optimized CPU production. It can provide manufacturers with the opportunity to create an optimal design, produce fewer

faulty products, and increase customer satisfaction. This will increase the competitiveness of these companies and lead to positive results for end-users and firms alike.

Materials and Methods

Data Gathering

There are several features that affect the performance of a CPU. At the top of these are features such as core speed and clock speed. The whole list of features is explained below.

1. Base Clock: Base clock defines the speed at which the processor's transistors open and close. Base clock is the operating point at which Thermal Design Power (TDP) is defined. It is calculated in Gigahertz (GHz) or cycle per second.
2. CPU Mark [5]: CPU Mark/CPU Point refers to the score of the processor after benchmarking tests. This will be output for our models.
3. Date: Release date of CPU.
4. Lithography [6]: It refers to the thickness of the processor's print on the silicon.
5. L3 Cache: It refers to part in the processor where fast memory is located.
6. Max. Boost Clock: Maximum boost clock is the maximum single-core frequency at which the processor can operate using various technologies, if available. It is calculated in Gigahertz (GHz) or cycle per second.
7. Number of Cores: The number of cores of the CPU. Cores are a hardware term that indicates the number of individual central processing units in a single computing component (chip).
8. Number of Threads: The number of threads of the CPU. Thread and thread processing are software terms for a simple regular sequence of instructions that pass through or are processed by a single CPU core.
9. Platform: Target platform of CPU.
10. Series: Series of CPU
11. TDP: TDP reflects the average power, in watts, that the processor dissipates at Base Clock with all active cores under a high complexity workload.
12. T_{jmax}: T_{jmax} is the maximum temperature allowed in the processor chip.

The CPUs in our collected dataset are the Intel [7] and AMD [8] branded CPUs, which are currently the two most commonly used brands. The data for these CPUs were collected from the official websites of the brands. The CPUs selected for use in the dataset have benchmark scores between 2749 and 63599. The statistical data related to the dataset is shown in Table 1. Although the range value of the date feature is 12, it can be seen that relatively new processors were selected by looking at the mean and mode values. The values of the base clock, max. boost clock, and series features did not vary much among the selected CPUs and remained in a narrow range. Although $Tjmax$ values are mostly clustered around the mean, the wide range indicates the presence of extreme values. The core and thread counts are also centered around the mean, but since the median values are lower than the mean and the range values are higher, we can conclude that they are more widely distributed. The output value, the CPU Mark value, is in a wide range and the variance value is quite high. In the light of all this data, it can be concluded from Table 1 that the slightest change in the input values can make a big difference in the CPU Mark value.

Methods

Data Collection and Pre-processing

Data mining in computer science is a field that involves the automated extraction of valuable patterns and knowledge from large datasets. It is an interdisciplinary subfield of computer science that utilizes techniques from artificial intelligence, statistics, and database management [9].

Data is considered to be of good quality if it meets the requirements of the intended use. Apart from this, there are three elements that define data quality: accuracy, completeness and consistency [10]. Different attribute types and data properties can help detect false values and outliers. This will be useful in data cleaning and integration steps. Data processing techniques, when applied before mining, can significantly improve the overall quality of the

Table 1. Statistical Data

Base Clock	CPU Mark	Date	Litography	L3 Cache	Max. Boost Clock	# of Cores	# of Threads	Platform	Series	TDP	Tjmax	
3.10	14743.98	2018.54	12.92	13.37	4.00	6.80	12.17	6.26	6.20	66.06	97.51	Mean
0.58	10648.6	3.11	5.69	10.64	0.75	3.75	6.92	1.70	1.89	45.92	6.81	STD
3.10	12714.00	2019.00	14.00	12.00	4.00	6.00	12.00	7	7.00	56.5	100.00	Median
3.00	16979.00	2022.00	14.00	8.00	4.00	4.00	8.00	7	7.00	65.00	100.00	Mode
3.20	63591.1	12.00	27.00	61.00	2.00	22.00	32.00	6	8.00	247.00	41.00	Range
0.33	1.13E+08	9.67	32.39	113.41	0.56	14.11	48.00	2.89	3.57	2109.44	46.41	Variance

extracted patterns and/or the time required for actual mining [11].

Data mining methods and techniques cover various areas such as classification, clustering, exploratory analysis, relationship analysis, prediction and text mining. These methods are used to transform complex data sets into meaningful information and improve decision-making processes.

The steps in the data mining process can be listed as follows:

1. Data Cleaning: The removal of meaningless data from the dataset.
2. Data Integration: The meaningful combination of different data.
3. Data Reduction: Determining the data to be used for obtaining meaningful information.
4. Data Transformation: Preparing the data to be applied to data mining, making it suitable for the data mining algorithm to be used.
5. Data Mining: Applying a machine learning algorithm to the dataset that has been prepared for data mining.
6. Pattern Evaluation: Identifying different patterns within the pattern obtained as a result of the used algorithm.
7. Knowledge Representation: Presenting the new information obtained as a result of data mining to the user.

The dataset used in this study has categorical data alongside numerical data. Platform and series features are encoded to representative numbers during preprocessing phase. Processors produced for the desktop platform tend to have higher benchmark scores than those produced for the laptop platform. Similarly, Series feature indicates the generation of CPU. That means the higher the series, the higher the performance tends to be.

Algorithms

The data, which has been made suitable for the use of machine learning algorithms, has been processed through regression analysis algorithms to predict performance scores. The algorithms used are as follows:

Support Vector Regression

The Support Vector Machine was presented in 1995 [12] and generalizes to Support Vector Regression (SVR) by creating an ε-tube, an ε-insensitive region around the function. SVR is a machine learning technique that involves training a model to predict a continuous output variable by finding a hyperplane that best fits the data. In support vector regression, the goal is to find a hyperplane that fits the data while allowing some points to be on the wrong side of the hyperplane, within a certain margin of error. These points are called support vectors, and they are used to define the hyperplane [13].

The SVR using a linear kernel function is shown in Equation 1 [14].

$$y = \omega x + b \tag{1}$$

The Error Function (Equation 2) can be minimized based on the target being z_i [14].

$$min\frac{1}{2}\|\omega\| + c\sum_{i=1}^{n}(\xi_j + \xi_i^*) \tag{2}$$

$$Subject\ to \begin{cases} z_i - (\omega x + b), & \leq \varepsilon + \xi_i \\ (\omega..x) + b - z_i, & \leq \varepsilon + \xi_i^* \\ \xi_i\xi_i^*, & \geq 0 \end{cases} \tag{3}$$

The linear kernel function can be described as follows [14]:

$$\mathrm{K(x, y)} = \langle x, y \rangle \tag{4}$$

Random Forest Regression

Random Forest Regression (RFR) is a machine learning technique that involves creating an ensemble of unpruned decision trees through the use of bootstrap samples of the training data and random feature selection during tree induction [15]. This method, first developed by Breiman in 2001 [16] and has been shown to outperform linear regression in terms of fitting observed data due to its ability to handle variance in the data more effectively [17]. It is capable of handling relatively small samples with a large number of variables, making it a robust choice for various applications [18].

The process of constructing a RFR model involves tuning several hyperparameters, such as the number of trees in the forest and the maximum depth of each tree. These hyperparameters can be optimized using techniques such as grid search or random search [19].

Multiple Linear Regression

Multiple Linear Regression (MLR) is a statistical technique used to analyze the relationship between multiple independent variables and a single dependent variable. It extends the concept of simple linear regression by allowing for more than one predictor variable to be included in the model [20]. In computer science, linear and multiple linear regression finds various applications such as in developing predictive models, analyzing sensor data, and calibrating sensor technologies [21][22]. MLR can be used to understand the relationship between CPU performance and different hardware components and other observable data.

The mathematical model of multiple linear regression can be explained as follows [23]:

$$y_i = \beta_0 + \beta_1 x_i + \cdots + \beta_p x_{ip} + \epsilon_i, \quad i = 1, \ldots, n \quad (5)$$

Here, y is the dependent variable and x is the independent variable. β_0 represents the constant, and β_n represents the coefficients.

Gradient Boosting Regression

Gradient boosting is a technique in ensemble learning where numerous weak learners, like decision trees, are merged to form a powerful learner [24][25]. In Gradient Boosting Regression (GBR), each decision tree is trained to predict the residual error of the previous tree, and the final prediction is obtained by adding the predictions of all the trees. This approach allows the model to focus on the most difficult cases and improve its performance over time.

The algorithm operates by iteratively incorporating decision trees into the model, [26], with each new tree trained to predict the negative gradient of the loss function with respect to the output variable. The negative gradient represents the direction in which the loss function is decreasing, and thus helps the algorithm to focus on the most difficult cases.

Neural Network Regression

Neural Networks was first proposed in 1943 and inspired by the biological nervous system [27]. Lately, they have become extensively used in classification, regression, pattern recognition, forecasting, and time series problems [28]-[30].

Neural Network Regression (NNR) is a method in machine learning where a neural network is trained to forecast a continuous output. A neural network comprises interconnected nodes, known as neurons, arranged in layers. The input layer receives data, while the output layer generates predictions. Intermediate hidden layers allow the network to grasp intricate non-linear connections between input and output variables. The objective in neural network regression is to minimize the disparity between predicted and actual outputs through training.

Neural networks can be tuned by adjusting several hyperparameters, such as hidden layer sizes, solver class and the learning rate of the optimization algorithm [31-37].

Feature Scaling

Min-Max Scaling

Due to the differences in scales between the CPU features in the dataset, it was decided to use data scaling. The Min Max Scaler method was applied for this purpose.

Min Max Scaler performs the scaling (normalization) of values in a dataset within a certain range. This is similar to standardizing the data, but while standardization scales the data based on the mean value, Min Max Scaler squeezes the data into a specified minimum and maximum range [38].

This can be useful, especially when there are many outliers in the dataset.

Min Max Scaler scales each feature in the dataset (which includes the attributes used by the learning algorithm) using the following formula:

$$x' = \frac{(x - \min(x))}{(\max(x) - \min(x))} \quad (6)$$

Here, x is the value of a feature in the dataset, and $max(x)$ and $min(x)$ are the maximum and minimum values of that feature. This formula scales the values of each feature in the dataset between 0 and 1 [38].

Evaluation Metrics

In this model, Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE) [39], and R-Squared (R^2) [40] metrics used to measure the difference between the predicted value and the actual value.

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |y_i - \hat{y}_i| \quad (7)$$

$$MSE = \frac{1}{n}\sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \quad (8)$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} (y_i - \hat{y}_i)^2} \quad (9)$$

$$R^2 = 1 - (\frac{\sum_j (t_j - o_j)^2}{\sum_j (o_j)^2} \quad (10)$$

Experimental Results & Discussion

In the collected dataset, 70% of the 206 data points were separated for training and 30% for testing, randomly.

Support Vector Regression

SVR results were the worst out of all five algorithms we compared. Results of each metric for SVR shown below.

- MAE: 2268.27
- MSE: 18808394.80
- RMSE: 4336.86
- R^2: 0.8658

In SVR, the hyperparameters that need to be tuned includes the choice of kernel function, gamma value and the regularization parameter that controls the trade-off between the margin and the error. These hyperparameters can be optimized using techniques such as grid search or random search. In our case, 'linear', '1' and '1000' values were the best fit for these parameters in order.

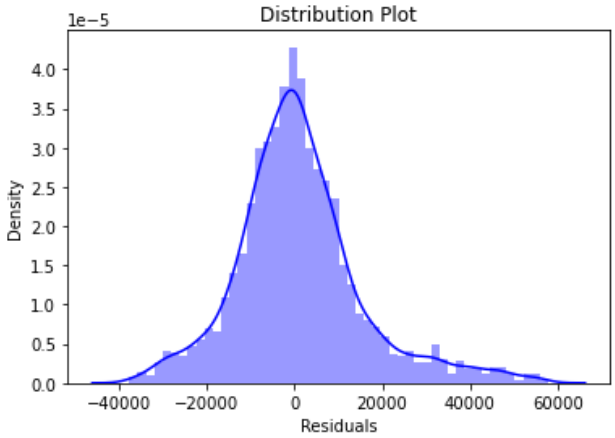

Figure 1. Distribution Plot of SVR.

Figure 1 shows the distribution of the residuals. Ideally the residuals should be symmetric around the zero line. If the residuals are skewed to one side, it suggests that the model is overpredicting or underpredicting the target variable in that region. We can see that residuals we have slightly skewed to right which means the model is not capturing some patterns in the data, or there may be outliers in the data that are affecting the model's performance.

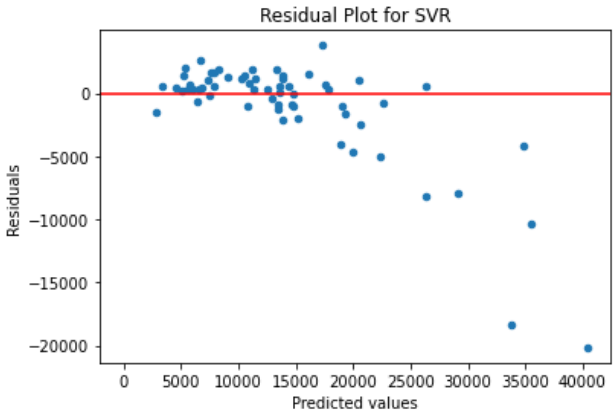

Figure 2. Residual Plot of SVR.

The Figure 2 appears to have a roughly asymmetric distribution around zero, which suggests that the SVR model is not making predictions that are on average equally overpredicting and underpredicting the true values.

It is clear that there are some outliers in the residual plot, particularly at the higher end of the x-axis, which suggests that the SVR model is struggling to accurately predict these extreme values.

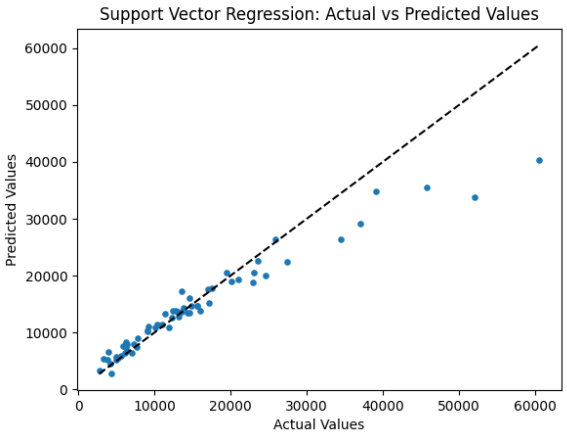

Figure 3. Scatter Plot of SVR.

Figure 3 shows us the scatter plot of the predicted and actual values. The dashed line represents the ideal prediction where actual and predicted values are the same. Points above the dashed line indicate that the model has over-predicted the target variable, whereas points below the line indicate under-prediction. The scatter plot can be used to identify areas where the model is performing well and where it needs improvement. For this instance, the SVR model heavily under-predicts values over 20000, making it unreliable for predicting high-end CPUs scores.

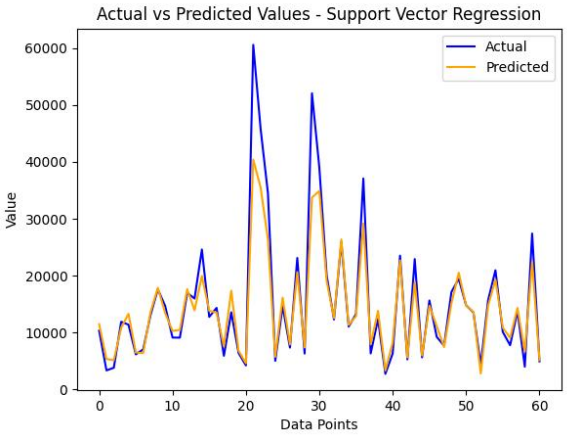

Figure 4. Actual vs Predicted Values of SVR.

This Figure 4 displays the comparison of the predicted and actual values of test CPUs. The blue line represents the actual values, while the orange line represents the predicted values. Ideally, these lines would overlap, with the predicted line covering the actual line, indicating that the model is providing accurate results. It is clear that the predicted values are significantly inaccurate for CPUs with score values of 20000 and above.

Random Forest Regression

RFR gave the second worst results out of all five algorithms we compared. Results of each metric for RFR shown below.

- MAE: 1828.75
- MSE: 9210826.03

- RMSE: 3034.93

- R^2: 0.9346

In RFR, after tuning several hyperparameters, we found that the number of trees parameter '100' and the function to measure the quality of a split 'squared error' gave better results than others we tried.

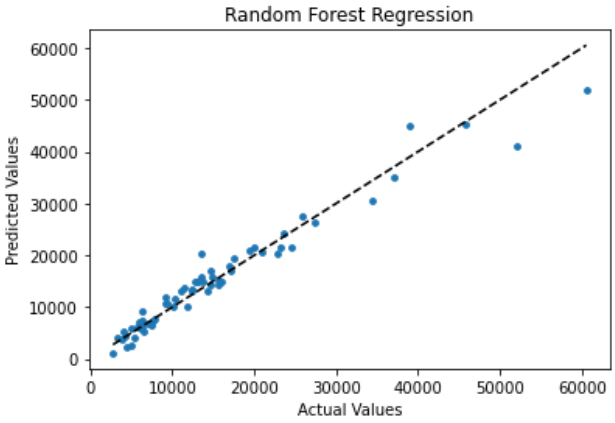

Figure 5. Scatter Graph of RFR.

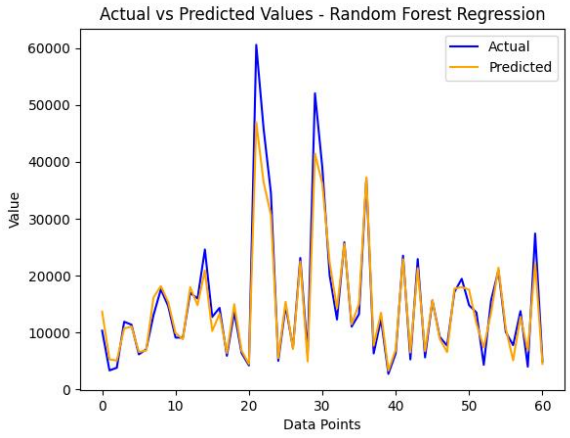

Figure 6. Actual vs Predicted Values of RFR.

Both Figure 5 and Figure 6 show us the comparison of predicted and actual values for their respective data points. It is clear that the RFR algorithm provides good results for lower values of CPU Score while still heavily under-predicting most values above 30000.

Multiple Linear Regression

MLR results showed that this algorithm was the third best among all the algorithms we compared. Results of each metric for MLR are demonstrated below.

- MAE: 1724.65

- MSE: 6701067.82

- RMSE: 2588.64

- R^2: 0.9522

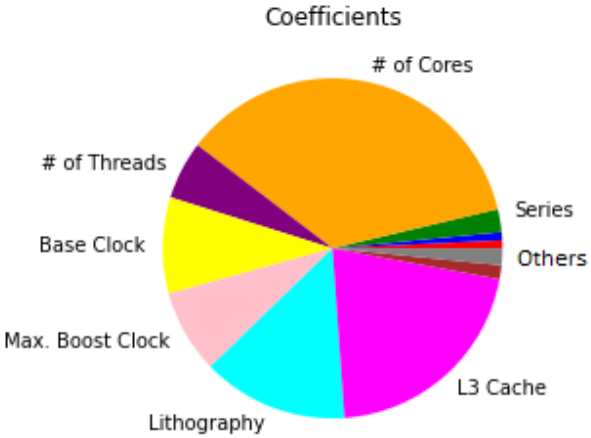

Figure 7. Coefficients Pie Chart of MLR.

Coefficient plots can be used to understand how each feature affects the outcome (Figure 7). In this pie chart, we can see the coefficients of 11 features that affect the target variable. According to the results, the most influential features are the number of cores and L3 cache, while the least influential ones are release date and platforms. Assuming that the features indicating the number of cores, CPU clock speed, L3 cache, and lithography of the CPU are also the most repeated and highlighted features in the introduction and advertisements of newly released CPUs, we can confirm that their impacts on the result are quite high as expected.

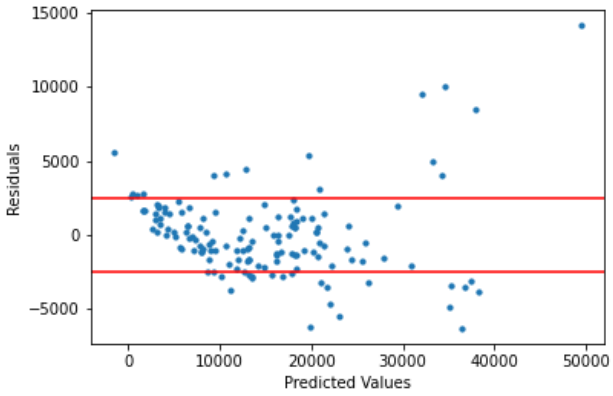

Figure 8. Residuals / Predictions Scatter Graph of MLR.

In Figure. 8, it is shown that the predicted values and their difference with actual values. Looking at the area between the +2500 and -2500 difference lines we drew, it can be understood that the absolute difference between most of the predicted values and the actual values is not more than 2500.

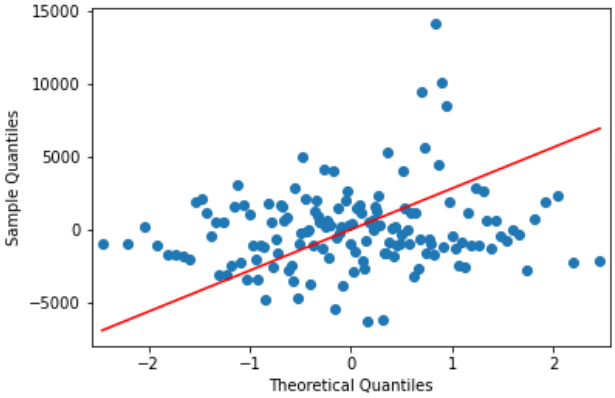

Figure 9. Q-Q Plot of MLR.

This Q-Q plot of MLR allows us to compare the status of our data's normal distribution with its theoretical quantity (Figure 9). The closer the data is to the center line, the closer the spread of the data is to a normal distribution.

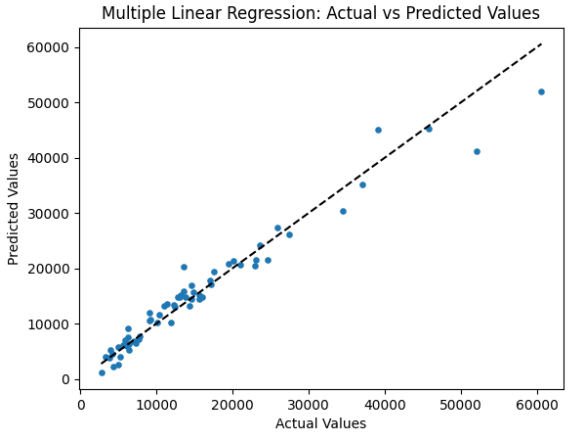

Figure 10. Scatter Plot of MLR.

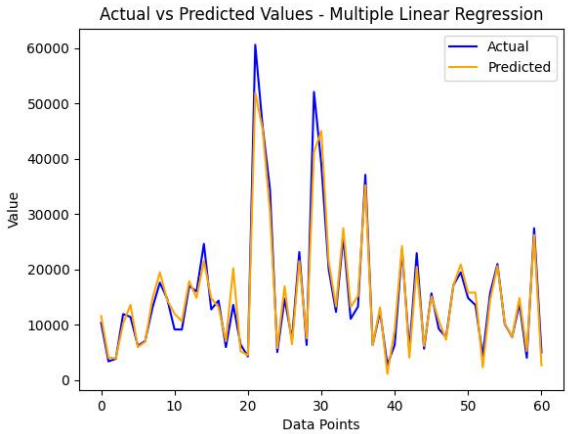

Figure 11. Actual vs Predicted Values of MLR.

Figure 10 and Figure 11 show that the model makes more mistakes in predicting CPU scores above 40000 on average. This may be due to the fact that these processors are relatively new, powerful, and rare, and there may not be enough data for them in the dataset for training.

Gradient Boosting Regression

GBR has the second-best results among all the five algorithms we compared. Results of each metric for GBR are shown below.

- MAE: 1608.57
- MSE: 5776080.55
- RMSE: 2403.34
- R^2: 0.9588

The process of constructing a gradient boosting regression model involves tuning several hyperparameters, such as loss function, the number of boosting stages to perform (n-predictors), the maximum depth of each tree, and the learning rate of the optimization algorithm. In our case, we found 'squared-error', '143', '1', '0.1' values were the best fit for these parameters in order.

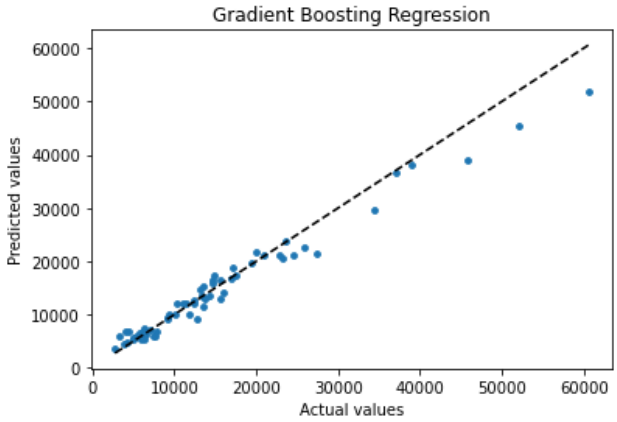

Figure 12. Scatter Plot of GBR.

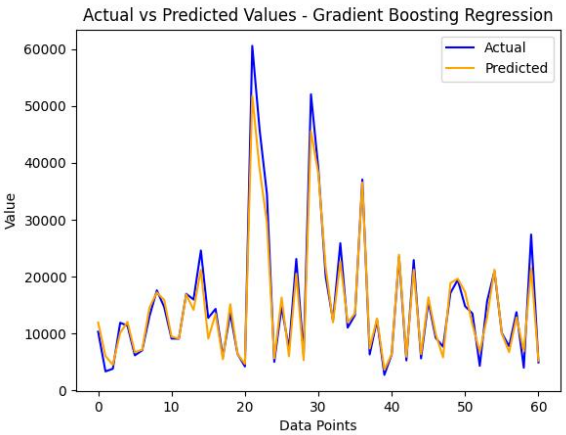

Figure 13. Actual vs Predicted Values of GBR.

Figure 12 and Figure 13 show the actual values of the target variable against the predicted values using GBR model for each data point. It is clear that this model's accurate prediction rate is higher than others we compared, which is also seen from metric results as well. We can see that the higher values of target variable show slightly under-predicted results, although not as much as the first three algorithms.

Neural Network Regression

NNR gives the absolute best results out of all the five algorithms that were discussed. Results of each metric for NNR are demonstrated below.

- MAE: 1245.61
- MSE: 3358348.46
- RMSE: 1832.57
- R^2: 0.9765

As mentioned earlier, neural networks can be tuned by adjusting several hyperparameters, such as hidden layer sizes, solver class, and the learning rate of the optimization algorithm.

Using hyperparameter tuning, it was found that '(10,)', 'lbfgs', and 'constant' values were the best fit for these parameters in order.

We used the L-BFGS optimization algorithm for our NNR model, as it provided better results compared to the other well-known solver 'Adam'. We chose L-BFGS because our dataset was relatively small, and L-BFGS is known to work well with small datasets. L-BFGS is a gradient-based method that uses limited memory to approximate the inverse Hessian matrix and update the weights in each iteration. It is particularly effective in handling non-convex problems, which are common in NNR [41].

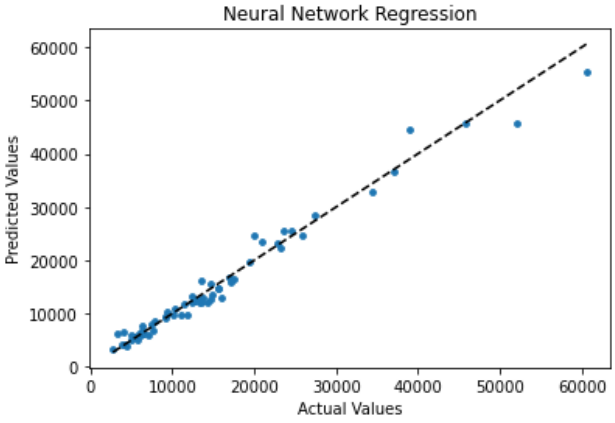

Figure 14. Scatter Plot of NNR.

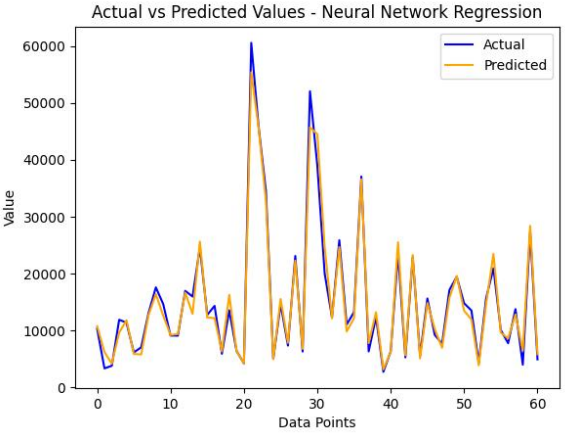

Figure 15. Actual vs Predicted Values of NNR.

NNR model gave the most accurate predictions, noticeably better than other algorithms. We know that the closer the points to the dashed line for Figure 14, and the more the lines overlap for Figure 15, the better the model's prediction performance. It is demonstrated that NNR has great accuracy on both lower values of CPU Score and higher values of CPU Score. Since dataset has more training data with lower values, the prior algorithms had a rough time to accurately predict, most of them ended up greatly under-predicting higher values, while NNR gave almost spot on predictions.

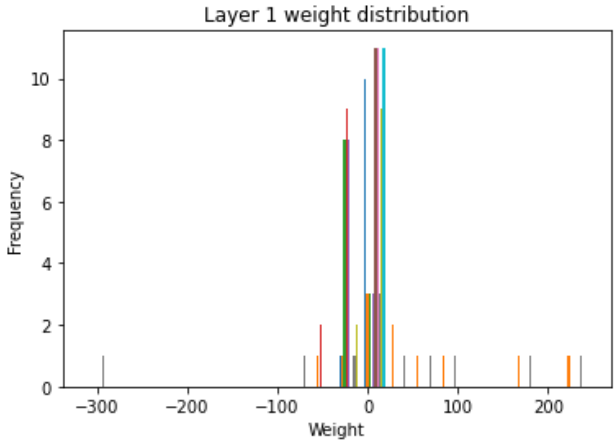

Figure 16. Histogram of Layer 1 Weight Distribution.

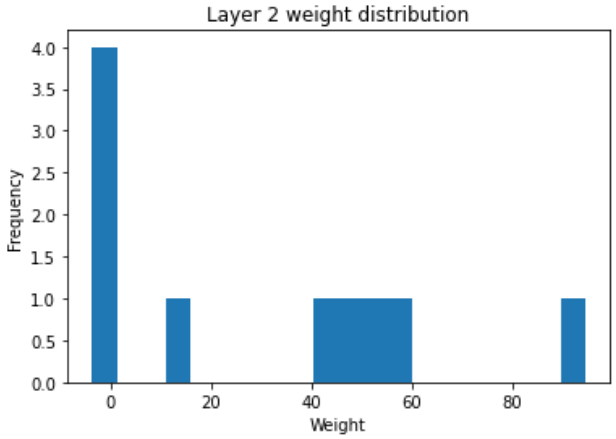

Figure 17. Histogram of Layer 2 Weight Distribution.

Figure 16 and Figure 17 are visual representations of the weights assigned to the nodes in a neural network model's layers. These graphs can be used to analyze how the weights change during training and to identify patterns or anomalies in the weight distribution.

The weights shown in Figure 16 and Figure 17 correspond to the connections between the input and hidden layers (layer 0 to layer 1) and the connections between the hidden and output layers (layer 1 to layer 2).

We can see on the layer 2 weight distribution Figure 17 that 4 out of 11 features have weights close to 0 on nodes, while 5 features have weight values of 40-60, and one feature has a weight value of more than 80. These results align well with what we have seen on coefficient charts, assigning

corresponding weight values to features that matter most like Number of Cores and Threads, L3 Cache, Clock Speeds, and features that matter least like Platform, Series, and T_{jmax}.

Comparison of these algorithms is shown in Table 2.

Table 2. Comparison of the results of the models used.

	SVR	RFR	MLR	GBR	NNR
MAE	2268.27	1828.75	1724.65	1608.57	1245.61
MSE	188083 94.80	9210826. 03	6701067 .82	5776080 .55	3358348 .46
RMSE	4336.86	3034.93	2588.64	2403.34	1832.57
R^2	0.8658	0.9346	0.9522	0.9588	0.9765

Conclusion

In this study, we attempted to predict the performance scores of CPUs ranging from 2749 to 63599 using regression analysis with a dataset consisting of 11 features. We evaluated several regression algorithms, including SVR, RFR, MLR, GBR and NNR.

Our results showed that the NNR algorithm achieved the best performance on the dataset. NNR achieved a value of 0.97 according to the R-Squared value. This result indicates that the NNR algorithm was able to capture the underlying relationships in the dataset more accurately than the other algorithms we tested. However, other algorithms such as MLR, RFR, GBR, and SVR also yielded moderate to good performance results, with R-squared values ranging from 0.8658 to 0.9588. These findings suggest that different algorithms may be suitable for different datasets, and the selection of the appropriate algorithm can have a significant impact on the accuracy of the predictions.

In conclusion, our study demonstrates the potential of using machine learning algorithms to predict the performance scores of CPUs. The NNR algorithm showed the best performance in our study, but the results from other algorithms also suggest that a variety of approaches should be considered when applying regression analysis to similar datasets. Future research could investigate the performance of these algorithms on larger or more diverse datasets, as well as the potential for further optimization of the algorithms themselves.

Furthermore, predicting a processor's benchmark score based on its features can be useful for evaluating performance without running benchmarks, saving time and resources. It allows users to make informed decisions about CPU's capabilities for specific tasks, such as gaming or data processing, by comparing predicted performance across models. This approach is particularly valuable for system builders and buyers looking to optimize cost and performance before purchasing.

Ethics committee approval and conflict of interest statement

There is no conflict of interest with any person / institution in the article prepared.

Authors' Contributions

-Study conception and design: Kaya & Şen

-Acquisition of data: Kaya & Şen

-Analysis and interpretation of data: Kaya & Şen

-Drafting of manuscript: Kaya & Şen & Altay

-Critical revision: Kaya & Altay

References

[1] J. Cao, J. Fu, M. Li, and J. Chen, "CPU load prediction for cloud environment based on a dynamic ensemble model," *Software, Practice & Experience/Software, Practice and Experience*, vol. 44, no. 7, pp. 793–804, Oct. 2013, doi: 10.1002/spe.2231.

[2] G. Contreras and M. Martonosi, "Power prediction for intel XScale® processors using performance monitoring unit events," Jan. 2005, doi: https://doi.org/10.1145/1077603.1077657.

[3] W. Wang, X. Huang, X. Qin, W. Zhang, J. Wei, and H. Zhong, "Application-Level CPU Consumption Estimation: Towards Performance Isolation of Multi-tenancy Web Applications," Jun. 2012, doi: https://doi.org/10.1109/cloud.2012.81.

[4] C. Foots, P. Pal, R. Datta, and A. Segev, "CPU Hardware Classification and Performance Prediction using Neural Networks and Statistical Learning," *International Journal of Artificial Intelligence & Applications*, vol. 11, no. 4, pp. 1–13, Jul. 2020, doi: https://doi.org/10.5121/ijaia.2020.11401.

[5] "Unbiased hardware comparisons," *technical.city*. https://technical.city/ (accessed Nov 25, 2022).

[6] "PassMark - CPU Benchmarks - CPU Mega Page - Detailed List of Benchmarked CPUs," *www.cpubenchmark.net*. https://www.cpubenchmark.net/CPU_mega_page.html (accessed Nov 25, 2022)

[7] "Intel® Processors for PC, Laptops, Servers, and AI," *Intel*. https://www.intel.com/content/www/us/en/products/details/processors.html (accessed Nov 25, 2022)

[8] AMD Processors | AMD. https://www.amd.com/en/processors/. [Accessed: Nov 25, 2022]

[9] L. Manikandan and R. Selvakumar, "A review on data mining concepts and tools", International Journal of Scientific Research in Computer Science, Engineering

and Information Technology, p. 600-605, 2022. https://doi.org/10.32628/cseit228683

[10] S. Garcia, J. Luengo, and F. Herrera, Data preprocessing in data mining. 2015. doi: 10.1007/978-3-319-10247-4

[11] J. Han, J. Pei, and H. Tong, "Data mining: concepts and techniques," Morgan Kaufmann, 2022.

[12] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, Sep. 1995, doi: https://doi.org/10.1007/bf00994018.

[13] M. Awad and R. Khanna, "Support Vector Regression," *Efficient Learning Machines*, pp. 67–80, 2015, doi: https://doi.org/10.1007/978-1-4302-5990-9_4.

[14] Kavitha S, Varuna S, and Ramya R, "A comparative analysis on linear regression and support vector regression," *2016 Online International Conference on Green Engineering and Technologies (IC-GET)*, Nov. 2016, doi: https://doi.org/10.1109/get.2016.7916627.

[15] A. Ali, R. Darvishzadeh, A. Skidmore, T. Gara, & M. Heurich, "Machine learning methods' performance in radiative transfer model inversion to retrieve plant traits from sentinel-2 data of a mixed mountain forest", International Journal of Digital Earth, vol. 14, no. 1, p. 106-120, 2020. https://doi.org/10.1080/17538947.2020.1794064

[16] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001, doi: https://doi.org/10.1023/a:1010933404324.

[17] E. Melišová, A. Vizina, M. Hanel, P. Pavlík, & P. Šuhájková, "Evaluation of evaporation from water reservoirs in local conditions at czech republic", Hydrology, vol. 8, no. 4, p. 153, 2021. https://doi.org/10.3390/hydrology8040153

[18] Yang, D., Li, S., Chong, Y., Li, M., Hu, W., Wang, B., … & Liu, P. (2020). Applying random forest model algorithm to gfr estimation.. https://doi.org/10.21203/rs.3.rs-22422/v1

[19] F. Hutter, Lars Kotthoff, and J. Vanschoren, *Automated machine learning : methods, systems, challenges*. Cham Springer, 2019.

[20] P. Pandit, P. Dey, & K. Krishnamurthy, "Comparative assessment of multiple linear regression and fuzzy linear regression models", SN Computer Science, vol. 2, no. 2, 2021. https://doi.org/10.1007/s42979-021-00473-3

[21] O. Altay, T. Gurgenc, M. Ulas, and C. Özel, "Prediction of wear loss quantities of ferro-alloy coating using different machine learning algorithms," *Friction*, vol. 8, no. 1, pp. 107–114, Jan. 2019, doi: https://doi.org/10.1007/s40544-018-0249-z.

[22] Thorson, J., Collier-Oxandale, A., & Hannigan, M. P. (2019). Using a low-cost sensor array and machine learning techniques to detect complex pollutant mixtures and identify likely sources. Sensors, 19(17), 3723. https://doi.org/10.3390/s19173723

[23] D. A. Freedman, *Statistical Models: Theory and Practice, -2nd ed.* Cambridge University Press, 2009.

[24] J. H. Friedman, "Stochastic gradient boosting," *Computational Statistics & Data Analysis*, vol. 38, no. 4, pp. 367–378, Feb. 2002, doi: https://doi.org/10.1016/s0167-9473(01)00065-2.

[25] C. Bentéjac, A. Csörgő, and G. Martínez-Muñoz, "A comparative analysis of gradient boosting algorithms," *Artificial Intelligence Review*, vol. 54, Aug. 2020, doi: https://doi.org/10.1007/s10462-020-09896-5.

[26] Y. Veisani, H. Sayyadi, A. Sahebi, G. Moradi, F. Mohamadian, and A. Delpisheh, "Comparison of machine learning algorithms to predict intentional and unintentional poisoning risk factors," Heliyon, vol. 9, no. 6, p. e17337, Jun. 2023, doi: https://doi.org/10.1016/j.heliyon.2023.e17337 .

[27] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The Bulletin of Mathematical Biophysics*, vol. 5, no. 4, pp. 115–133, Dec. 1943, doi: https://doi.org/10.1007/bf02478259.

[28] Z. Feng and W. Niu, "Hybrid artificial neural network and cooperation search algorithm for nonlinear river flow time series forecasting in humid and semi-humid regions," *Knowledge-Based Systems*, vol. 211, p. 106580, Jan. 2021, doi: https://doi.org/10.1016/j.knosys.2020.106580.

[29] A. Ameri, M. A. Akhaee, E. Scheme, and K. Englehart, "Regression convolutional neural network for improved simultaneous EMG control," *Journal of Neural Engineering*, vol. 16, no. 3, p. 036015, Apr. 2019, doi: https://doi.org/10.1088/1741-2552/ab0e2e.

[30] V. W. Y. Tam, A. Butera, K. N. Le, L. C. F. D. Silva, and A. C. J. Evangelista, "A prediction model for compressive strength of CO2 concrete using regression analysis and artificial neural networks," *Construction and Building Materials*, vol. 324, p. 126689, Mar. 2022, doi: https://doi.org/10.1016/j.conbuildmat.2022.126689.

[31] H. Hristov and G. Momcheva, "Hyperparameter adjustment in regression neural networks for predicting support case durations," *AIP conference proceedings*, Jan. 2021, doi: https://doi.org/10.1063/5.0041936.

[32] L. N. Smith, "A disciplined approach to neural network hyper-parameters: Part 1 -- learning rate, batch size, momentum, and weight decay," *arXiv.org*, Apr. 24, 2018. https://arxiv.org/abs/1803.09820v2 (accessed Nov. 17, 2023).

[33] Gurgenc, E., Altay, O., and Altay, E. V. (2024). AOSMA-MLP: A Novel Method for Hybrid Metaheuristics Artificial Neural Networks and a New Approach for Prediction of Geothermal Reservoir

Temperature. Applied Sciences, 14(8), 3534. doi: https://doi.org/10.3390/app14083534

[34] Altay, O., and Gurgenc, T. (2024). GJO-MLP: A Novel Method for Hybrid Metaheuristics Multi-Layer Perceptron And A New Approach For Prediction Of Wear Loss Of Az91d Magnesium Alloy Worn At Dry, Oil, And H-Bn Nanoadditive Oil. Surface Review and Letters (SRL), 31(06), 1-16. doi: https://doi.org/10.1142/S0218625X24500483

[35] Altay, O., and Varol Altay, E. (2023). A novel hybrid multilayer perceptron neural network with improved grey wolf optimizer. Neural Computing and Applications, 35(1), 529-556. doi: https://doi.org/10.1007/s00521-022-07775-4

[36] Altay, E. V., Gurgenc, E., Altay, O., and Dikici, A. (2022). Hybrid artificial neural network based on a metaheuristic optimization algorithm for the prediction of reservoir temperature using hydrogeochemical data of different geothermal areas in Anatolia (Turkey). Geothermics, 104, 102476. doi: https://doi.org/10.1016/j.geothermics.2022.102476

[37] Gurgenc, T., and Altay, O. (2022). Surface roughness prediction of wire electric discharge machining (WEDM)-machined AZ91D magnesium alloy using multilayer perceptron, ensemble neural network, and evolving product-unit neural network. Materials Testing, 64(3), 350-362. doi: https://doi.org/10.1515/mt-2021-2034

[38] L. A. Shalabi, Z. Shaaban, and B. Kasasbeh, "Data Mining: A Preprocessing Engine," *Journal of Computer Science*, vol. 2, no. 9, pp. 735–739, Sep. 2006, doi: https://doi.org/10.3844/jcssp.2006.735.739.

[39] C. Willmott and K. Matsuura, "Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance," *Climate Research*, vol. 30, no. 1, pp. 79–82, 2005, doi: https://doi.org/10.3354/cr030079.

[40] D. Chicco, M. J. Warrens, and G. Jurman, "The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation," *PeerJ Computer Science*, vol. 7, no. 5, p. e623, Jul. 2021, doi: https://doi.org/10.7717/peerj-cs.623.

[41] Y. Shao, F. M. Dietrich, C. Nettelblad, and C. Zhang, "Training algorithm matters for the performance of neural network potential: A case study of Adam and the Kalman filter optimizers," *Journal of chemical physics online/The Journal of chemical physics/Journal of chemical physics*, vol. 155, no. 20, Nov. 2021, doi: https://doi.org/10.1063/5.0070931.