Analysis of the Summit Login Nodes Usage Data

Shubham Rastogi
August 14, 2022

Introduction

This report presents my solutions to data challenge #2 posed to programmers this year involves conducting an in-depth analysis of the usage patterns for each of the Summit supercomputer's login nodes, and determining the solutions to related questions. This was completed through the organization of the dataset into the appropriate format (along with the categorization of data based on monthly intervals) and an analysis of usage patterns using statistical methods. Statistical prediction models and python programming can reveal surges in usage at one moment and drops in another moment- patterns often influenced by numerous external variables. Information on chronological changes in usage behavior can prove helpful for the administrators when allocating resources for maximum efficiency.

This challenge consists of four distinct problems involving one massive set of data. Problem 1 is to create an optimized csv file(s) from the dataset to simplify the parsing of data and increase the overall efficiency of analysis in later steps. Problem 2 is to draw a correlation between the usage of individuals and the progress of weeks. For instance, lower user density is expected during the weekends when compared with business days, and such a correlation can be identified and supported by lower usage values on particular days (i.e. Saturday or Sunday, in this example). To visualize a correlation between individual users, memory-usage, and total number of running processes; I created line graphs containing the forenamed variables for both of the problems. Problem 3 requires us to identify the usage of five login nodes with respect to specific parameters or dimensions such as unique users, memory-used, running-process, running-jobs and cpu_usage percentage. This will help in comparing and analyzing patterns of the usage parameters among the five login nodes. Problem 4 is to predict future trends. I used linear regression to predict the usage patterns of the login nodes for this purpose.

Proposed solution for Problem 1 *Capture the data and organize it into one or more easily usable CSV datasets.*

The solution to the first problem involved organizing the dataset into a format that is easier to read with the intention of making the data more accessible and convenient to process. The original dataset, containing 24 months worth of daily usage information of five different login nodes was in a plain text format bounded by the command name and ended with <'end'+command name>. I created a file pointer to read, filter, and organize the data into a list that gets appended to a dictionary with respect to its key values. Those key values serve as the columns for the *csv* datasets. I processed the total number of unique users, memory usage values, "running-process", "cpu_usage %", "running-jobs", "total-jobs", "time-to-ls", "time-for-1G", and "df-util"; and stored those values in a dictionary prior to converting them into a dataframe, and then to a *csv* file. Then I used *pandas*, a python library, to create *csv* datasets from the dictionary. I used the method `to_csv()` to create a *csv* file of that month after processing the data/ filtering the data from the DataSet. The month-wise files are created and named in the format <Month_Year .csv>. To clarify, the data for the month of May in the year 2020 is saved as a *csv* file named 'May2020.csv'. These files include information on the usage of operational login nodes of Summit for every single day spanning 2020-2021, with an exception of a few missing data.

The *csv* dataset is convenient as it lets us analyze the monthly details of the usage of login nodes for the span of 2 years. The strategy for this solution to use the pandas library eases our work of creating a *csv* file than the standard *csv* module to create a *csv* file. Creating a *csv* file by using dataframe and the functions of pandas rather than using the csv library is more efficient.The sample output of this methodology is shown in Table 1.

Table 1 *Sample of csv Dataset*

date-hour	login-node	wusers	memory-used	running-process	cpu_usage %	running-jobs	total-jobs	time-to-ls	time-for-1G	df-usage
Apr07_2020-00	login2	75	500.56	2104	832.0	75	488	0m0.003s	0m23.959s	905
Apr07_2020-01	login2	74	502.85	2126	1356.0	38	420	0m0.004s	0m23.978s	905
Apr07_2020-02	login2	64	375.77	2023	72.0	23	470	0m0.003s	0m24.121s	905
Apr07_2020-03	login2	60	375.9	2020	96.1	42	464	0m0.003s	0m24.983s	905
Apr07_2020-04	login2	59	375.55	2021	79.5	94	442	0m0.003s	0m23.988s	905
Apr07_2020-05	login2	60	376.48	2040	196.1	83	410	0m0.003s	0m24.042s	905
Apr07_2020-06	login2	59	376.46	2039	168.0	39	444	0m0.003s	0m23.898s	904
Apr07_2020-07	login2	59	376.43	2042	162.7	20	304	0m0.003s	0m23.951s	905

The simplicity of the data in the *csv* format is easier to comprehend and analyze. We can use the *csv* dataset to obtain any kind of information like usage of nodes in hour-00 for a day or

usage of nodes on a particular date ‘Apr05_2020’ etc. We can create a single *csv* dataset containing all the information of 2 years. But that would become too complicated and difficult to read.

The *csv* files are created in a folder named ‘*csv* files’ like ‘May2020.csv’, ’Oct2021.csv’. The forenamed name ‘May2020’ indicates that the files contain the data of the month May and year 2020. The csv files for the timeline from ‘Jan2020’ to ‘Dec2021’ will be created and organized in the folder. We can easily look at the data and get a deeper understanding about the timeline of the data record. This brings us one step closer to connecting usage patterns with outside variables and its relationship with worldly events, such as the COVID-19 pandemic.

Proposed solution for Problem 2 *Correlate the usage patterns with external events such as weekly working and non-working days such as weekends and public holidays (a list of such days will be provided)*

This problem has two parts -1) To show the correlation between the weekdays v/s weekend usage and 2) To show the correlation between the working v/s non-working hours.
To show the correlation of the weekdays v/s weekend I filtered the data with the timeline starting from Jan2020, and ending in Dec2021. Then I separated the data into 2 categories: weekends and weekdays. To be able to show the correlation we have to make sure that the data is of equal length. So, I created a week taking the ratio of weekdays and weekends in 5:2. The leftover days were ignored. I plotted the usage of 5 weekdays to 2 days of weekend and it showed a pattern. I proposed 3 different correlations of the dimensions namely unique users, memory-usage and running-processes. I got 3-4 weeks from a month considering the exclusion of the missing files, due to maintenance days, outages or any other unforeseen situations.

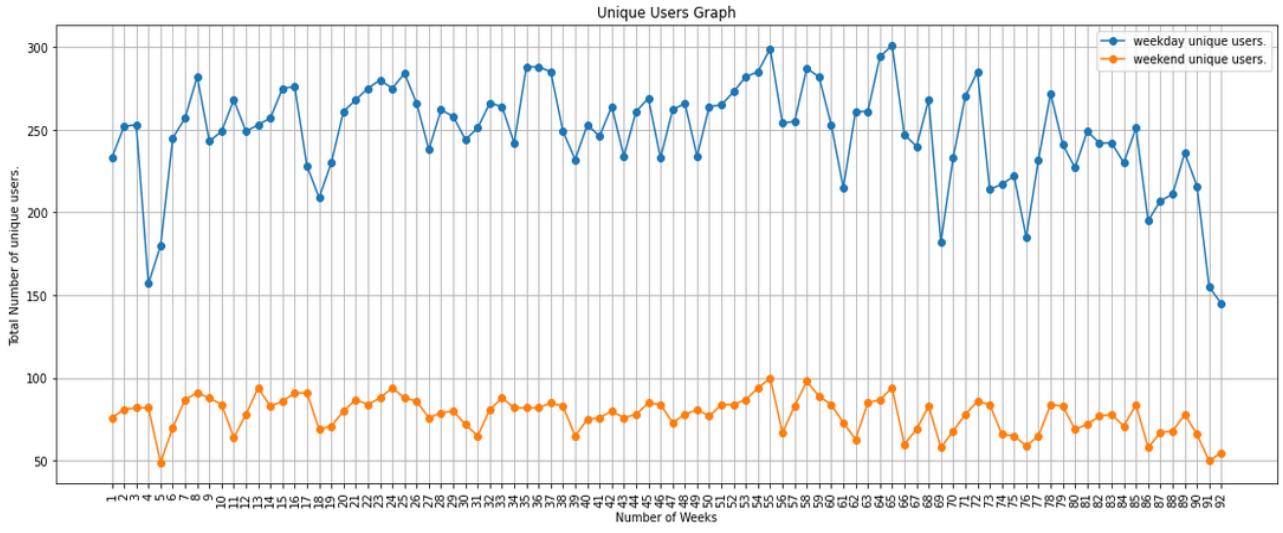
The correlation graph of working v/s non working hours can be created in a manner similar to Part 1. For part 2, I took into consideration that a good comparison can be achieved by taking into account the weekdays only. So, I separated the data of weekdays as working hours versus non-working hours. The data is not continuous for all the days and some of the files are missing and certain files where summit is operational for even less than 12 hours. In order to skew my calculations, that data is skipped. The data is filtered in 2 lists of working and non working hours. To plot the graph I’ve to make the data of equal length. However, working hours are 9 am to 5 pm and the rest are non-working hours leading to unequal lengths. I took the average with respect to the operational data in the lists to make a singular value for a day. Same process is done for the whole 2 year timeline. Then I plotted the correlation for the graph for unique users, memory-usage, and running-processes.

The data is fetched from the *csv* files which I created for solution of Problem 1. The csv data set is filtered according to the criteria whether it's a weekday or a weekend/holiday from the list provided by the organizers. After filtering, I took the ratio of data of 5 weekdays : 2 weekends/holidays (creating a week to plot on the graph for a weekly usage). Similar process is done in the working v/s non working graph where the data is first filtered considering that it is a weekday and the summit is operational for at least 12 hours. The data is stored in 2 lists containing working and non-working hours. Then by considering 5 weekdays(for a week) the graph is plotted using both the lists of working hours and non-working hours. I plotted the graph for the dimensions of unique users, memory usage and running processes of working and non-working hours.

The uniqueness of this approach is that first I separated the weekday and weekend/holidays from the *csv* dataset from Problem 1. I separated the weekdays and weekend/holidays from the data for further plotting to show the correlation. I showed the correlations for the dimension like unique users, memory-usage, and running process. Similarly, I did for the working vs non working hour graphs. I took the data of working days and filtered it in working hours vs non working hours to show a correlation or a pattern in a graph which is helpful in understanding easily. So, I plotted a graph for better understanding of the usage and showing correlation between them.

As mentioned above, I plotted the graphs for the dimensions to show the correlation. The results which we have got are the total number of unique users logged in on those 5 weekdays vs that weekend/ holidays of that week. Figure 1 demonstrates that the pattern of the weekdays is higher at the time of COVID-19 and other factors than of the weekend/holidays.

Figure 1 *Weekly Correlation of users on weekdays v/s weekends*



It also shows the varying usage of weekdays in comparison to the not so varied usage of weekends. The weekdays are affected by the external entities and so the usage shows many ups and downs.

Figure 2 *Weekly Correlation of running-processes on weekdays v/s weekends*

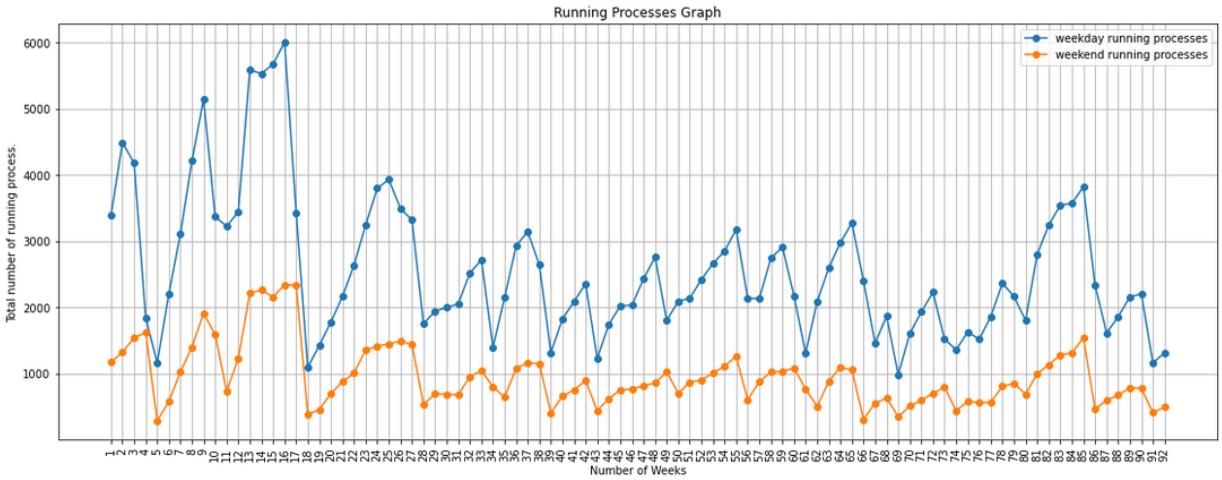


Figure 2 shows the varying usage of both weekdays and weekends in the month of Apr2020. It indicates that the running-process dimension was clearly affected for the whole week and the weekends are showing many ups and downs throughout the timeline too.

Figure 3 *Weekly Correlation between the working v/s non-working hours*

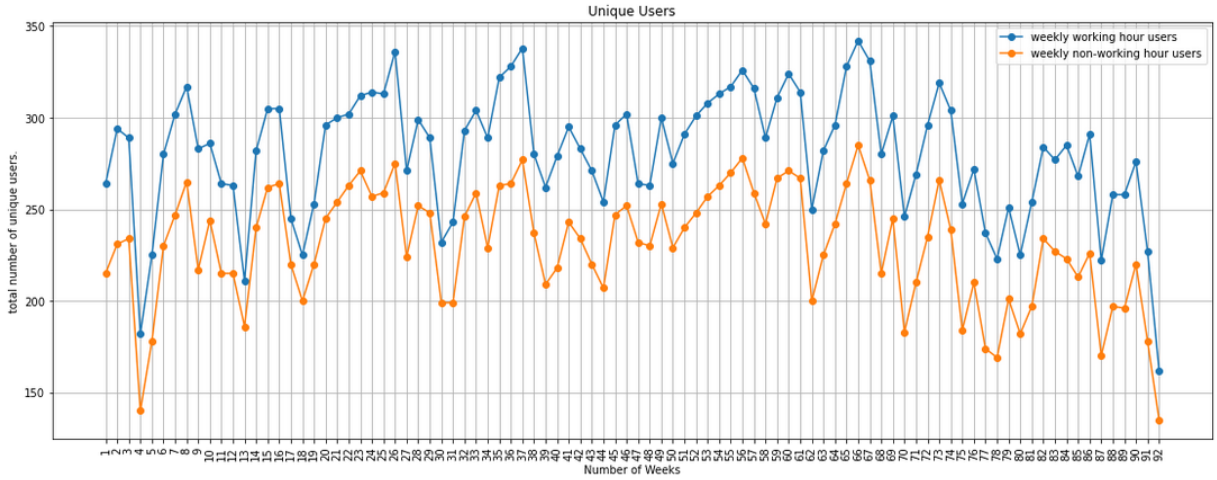


Figure 3 shows the correlation between working v/s non working hours graph. It shows the spikes throughout the timeline at all the operational hours.

Figure 4 *Weekly Correlation between the working v/s non-working hours*

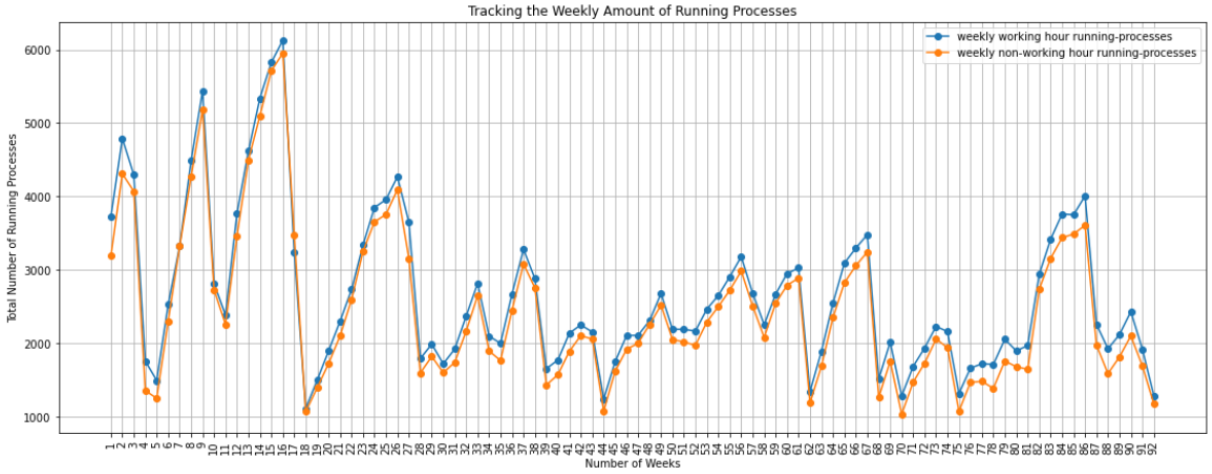


Figure 4 shows the pattern of the data generated according to the usage of the operational nodes for weekly working hours v/s the weekly non-working hours. It gives the idea of the average usage of running-processes on business days.

Proposed solution for Problem 3 *Understand and identify the relative system usage and any possible skews between the five login nodes. Derive a "state" defined by the various usage parameters and compare the state among the five login nodes.*

As my solution to Problem 3, I used a spider chart of the individual login node usage with respect to the dimensions like unique users, memory-usage, running-process, cpu_usage, running-jobs. I've tried to create a dictionary of the login nodes where each node has these dimensions to fetch the data of each node from the *csv* data set. I filtered data of all the days where the nodes were operational to identify an average usage of login nodes to their dimensions. I plotted a spider chart / radar chart to give a pictorial idea about the usage of the nodes wrt to the dimensions. I took the monthly average of usage of nodes and then the average of the timeline for all the nodes. The monthly data is processed from the *csv* dataset (of Problem 1), that data is appended to another dictionary which stores the average monthly data. Final matrix is prepared from this dictionary by taking the average of the values having values for the dimensions to the individual login nodes. The matrix has some outliers and some smaller values which are difficult to plot and understand. So, I took an equal proportion of the values and plotted them in the graph for better understanding. The graph even shows the high to low wrt to the dimensions of the 5 login nodes. This would help to identify the actual usage of the nodes individually wrt to other node dimensions.

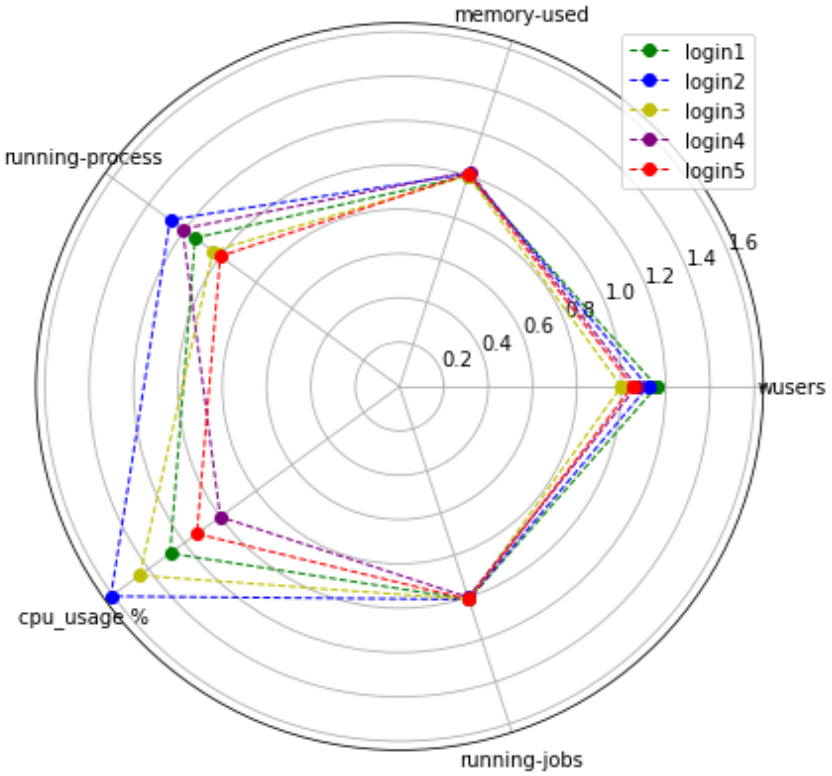
The data is filtered and processed for an operational day of the month and is appended to a dictionary having key values of login nodes. Then, the values are appended to the final dictionary, after the calculation for a month is over. The final dictionary will have the values of the average usage of the login nodes for the timeline. So, to get a final matrix with only one value for a node and its dimension, I took the average of the timeline and stored it in the final dictionary. The final dictionary will have a few outliers like memory-usage and running-processes. So, to plot them in equal proportion, I divided the values with their respective dimension's minimum values. That gives the equal proportion of the values and then plotted on the spider chart/ radar chart.

The data is filtered for all the days and averaged on a monthly basis and that data is appended to the other dictionary. The average of the 24 months is used to create a final matrix having 25 values (5 login nodes and 5 dimensions.).That data consists of outliers and to make the graph look proportionate, I divided the values with the dimension's minimum values. The resultant proportionate values are plotted in the radar chart. I imported the matplotlib library to plot the radar chart and the pandas library to display the matrix and show the values of the dataframe.

I chose this unique approach because our goal was to find the individual login node usage to find the pattern between the high to lows wrt to other dimensions. A radar chart would be an easy to understand solution if we want to understand the usage of the nodes individually to their dimensions. The approach is to get the average of the data of the usage of the login nodes on a monthly basis and again take the average to get the final values . I handled the outliers, by taking equal proportion and then used the values for plotting. The resultant data is plotted on the graph for the better understanding of the interesting usage trend of the login nodes.

The result of the proportionate graph is in Figure 5 and is easy to identify the high wrt to the lows of login nodes to other dimensions.

Figure 5 *Spider chart for analyzing the login node's usage with respect to the dimensions*



Proposed solution for Problem 4 *Predict the future trends. Identify interesting usage trends and predict them with respect to each other with a given degree of confidentiality.*

To predict the usage and interesting trends, I used the regression technique (LinearRegression library in Python) and also plotted the prediction values over the graph. This led to an easier understanding of the usage trends and gave an insight into its behavior including it being strictly affected by any external events. I used the linear regression technique. The solution shows the average monthly usage during business days.

I filtered the data from the *csv* files, including business days to process. The data is averaged on a daily and then monthly basis. I plotted the models for 3 dimensions namely unique users, memory-usage and running-process. Then I used the regression library to predict the values from the regression model for that dimension. The graph plots two values - average monthly usage and the prediction values for the dimensions mentioned above.

I used this approach to find the averages on a daily and then monthly basis because I observed a pattern in the graphs of correlation of weekdays/weekends. The business days showed the spike which I thought would be due to certain external events including COVID-19, and vaccination announcements etc. To testify the theory, I used matplotlib to plot the graphs of average monthly usage and then used LinearRegression() to plot the prediction model along the graph.

The prediction models for the unique users, memory-usage and running-processes are as follows:

Figure 6 *Prediction model of unique users (monthly)*

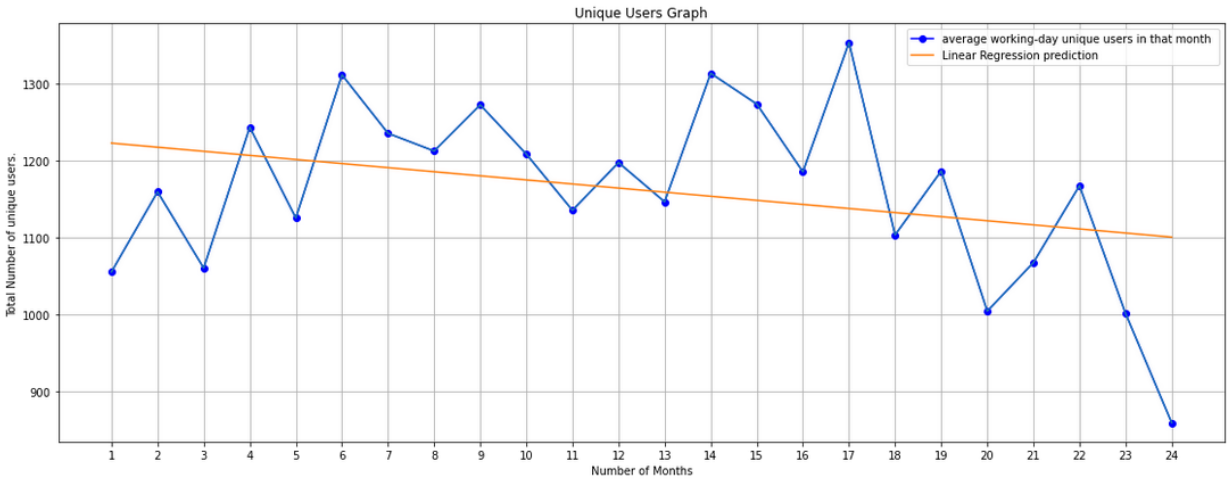


Figure 6 clearly indicates during the covid first wave (Apr2020) and the second wave the users usage were high. Even other events might also be involved for the high usage other than covid like vaccine announcements, unusual weather etc.

Figure 7 *Prediction model of memory-usage (monthly)*

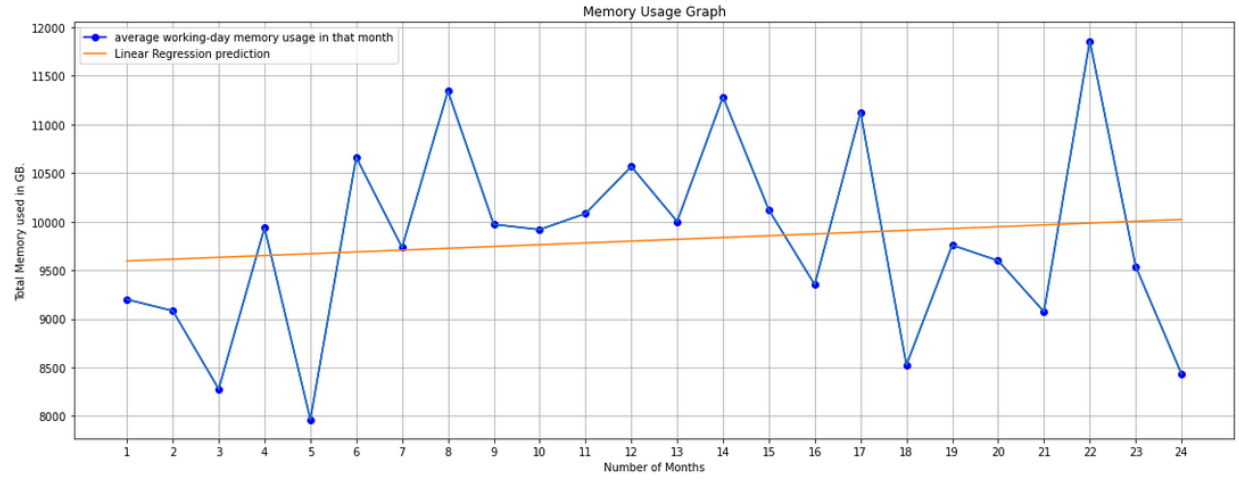


Figure 7 shows the increasing usage for processing from the fourth month (April 2020) that might indicate it being strictly altered by the external entities.

Figure 8 *Prediction model of running-processes (monthly)*

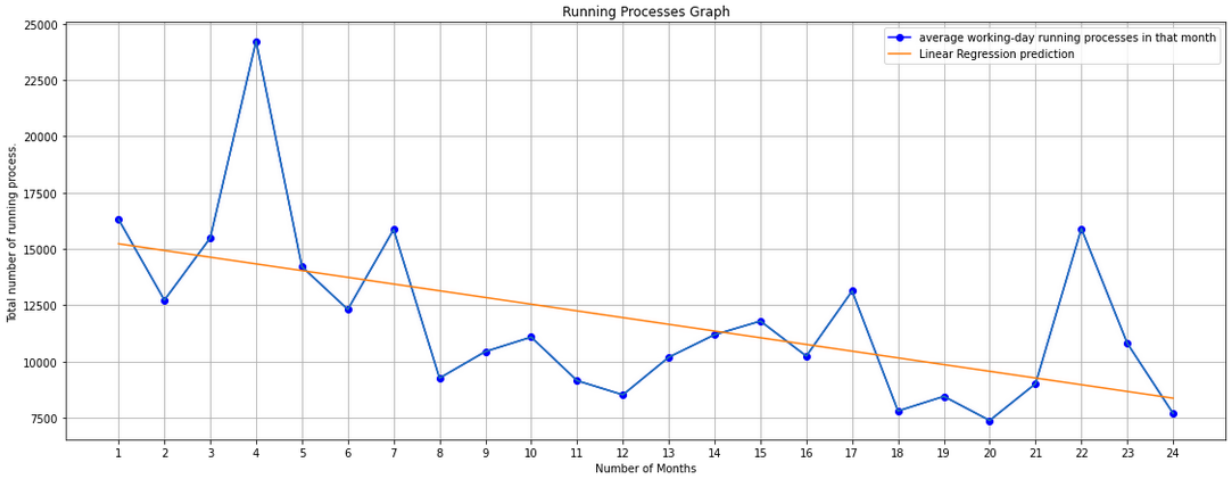


Figure 8 shows the graph for running processes the high usage compared to the prediction model is also a proof for supporting the theory of the increase in usage during the involvement of external entities. And It will even increase with the involvement of external entities in future.