Student ID: 201034888
Date: 02/10/2020

COMP702 – Energy Efficient Computing Project, Dissertation

By: Ryan Llewellyn

Supervisor: Dr Michael Bane

Secondary Supervisor: Dr Thanh-Toan Do

1. **ABSTRACT**:

The aim of this project was to minimise the energy consumption of a computer program over its runtime, thereby increasing the energy efficiency of computing.

We examine the feasibility of using machine learning to minimise the energy consumption of any given computer program over its runtime. By successful application of this approach we could therefore increase the energy efficiency of computing generally.

Our approach involves measuring the energy consumed during the run time of a given simulation, for a known set of user variables (e.g. compiler, computer chip architecture, number of parallel processing elements). We also require analysis of code characteristics such as the number of floating-point operations and run time (as explained further in Design Section).

For appropriate machine learning predictions, we require a vast set of these inputs and the corresponding energy consumption data. We therefore set out to construct an appropriate benchmarking suite.

Optimising for energy efficiency requires analysis of code characteristics and measurement of energy consumption for a variety of computer programs with different characteristics, to achieve this a benchmarking suite was created consisting of a variety of pre-existing benchmarks to allow for the simulation of common HPC workloads.

Through simulation of common High-Performance Computing (HPC) workloads this project aimed to gather characteristics common to typical HPC workloads and use said characteristics in training a machine learning model to provide an accurate prediction of energy consumption for an HPC computer program given its characteristics. This prediction is then used to suggest the optimal user configurable settings to minimise energy consumption.

For the purposes of this project all testing was completed using the Barkla HPC system at the University of Liverpool, The Barkla HPC system [1] uses the CentOS operating system based on the Linux kernel along with the Slurm workload manager.

Recording of energy consumption and code characteristics was accomplished using the Perf [2] tools program for Linux and the Benchmark suite consists of programs written in the C programming language to remove possible variance in energy efficiency of different programming languages from this projects results.

Student ID: 201034888
Date: 02/10/2020

Accomplishing the aims of this project a software suite was required consisting of several utilities:

- Slurm output parser
    - Reads benchmark output data and generates a comma separated values dataset.

- a dataset splitter
    - Divides a given dataset into training and testing datasets.

- a machine learning program
    - Requires training and testing datasets as input, perform feature selection and fitting of a machine learning model.
    - Makes a prediction using the testing dataset, also calculates an optimal user configuration of settings for a given workload.

The Benchmark suite consists of Benchmarks from 3 suites, the NASA Parallel Benchmarks suite, HPC Challenge Linpack benchmark and Mantevo Mini-Apps benchmark suite. [3, 4, 5]

The project produced a proof of concept system for the running of the benchmark suite, use of the software suite to analyse outputs and generate a prediction of energy consumption for given test data including suggested optimisations in the form of suggested user configurable options.

2. **INTRODUCTION**:

Overview & problem addressed: Workloads in a High-Performance Computing (HPC) environment can have runtimes lasting over several days or weeks, as a result of this energy consumption for HPC workloads can be high. Despite this, it is not common for users to attempt to determine the optimum user configurable settings to minimise energy consumption for a given workload, since it can be time consuming and involve much trial and error. The aim of this project is to take a workload and based on the characteristics of its code, generate a predicted energy consumption value and suggest optimal user configurable settings to minimise energy consumption. By automating this process, we would expect greater uptake by users and administrators of supercomputing facilities, thus resulting in significant reduction of energy consumption and thus carbon emissions and financial cost.

Student ID: 201034888
Date: 02/10/2020

High Performance Computing typically has 3 limitations with regards to a workload:

- Computing resources available: A given HPC system has a finite amount of computational resources to run a workload and most HPC systems are multi-user meaning there is more than 1 workload to be run on the HPC system, therefore it is ideal to use the minimum computational resources possible to successfully execute a given workload.
- Time: Some workloads can be time sensitive for example the results from a weather prediction workload are only valuable for a specific time period in the future, HPC systems can also charge per minute of operation for a customer's given workload, therefore it is ideal to minimise the time a workload takes to successfully execute.
- Energy: As a combination of computing resources used and time taken for workload execution, the energy consumption of a given workload is desired to be minimised for considerations of electricity cost for HPC operation during the execution of a workload and also the reduction of electricity consumption for environmental concerns to reduce carbon emissions from electricity generation.

This project aimed to maximise energy efficiency for typical workloads on an HPC system, through minimising energy consumption the aim is to reduce electricity consumption and therefore cost of HPC operation as well as reduce the carbon footprint of HPC.

Solution produced: The solution created by this project to minimise energy consumption consists of 2 suites of software, more details can be found in the design section:

1- Benchmarking suite, consisting of existing well-known HPC benchmarks which simulate common workloads. The suite consists of the following benchmarks:
    - NASA Parallel Benchmarks
    - HPC Challenge Benchmark
    - Mantevo Mini-Apps Benchmarks

2- Software suite, written during this project consisting of:
    - Slurm output parser and comma separated value dataset generator, taking data from benchmark results and gathering data on code characteristics and energy consumption.

    - train/test splitter to split a CSV dataset into training and testing datasets.

    - Machine Learning program
        i. Perform feature selection using 3 feature selection techniques:
            - Pearson's Correlation Coefficient
            - Spearman's Rank Coefficient
            - Kendall Rank Coefficient

- Fit machine learning model to dataset and features, consisting of two user selectable models, details of which can be found in the design section:
  - Linear Regression
  - Random Forest Regression

- Predict energy consumption value for each test case given program characteristics and use of this prediction to suggest optimal user configurable settings to minimise energy consumption.

User configurable settings: Refers to Slurm job scheduler arguments and compiler options, for the purposes of this project such settings were reduced to: (Note: example values)

1. Compiler used: GCC / ICC
2. Compiler optimisation level: O0, O1, O2, O3
3. Parallelisation technology: MPI, OpenMP, Hybrid (MPI + OpenMP)
4. Architecture: x86, ARM
5. Platform: Intel, AMD, ARM
6. Processor: Intel Xeon Gold 6138
7. Processor Generation: Skylake

Effectiveness of Solution: Solution produced allowed for parsing of benchmark output data to retrieve energy consumption and code characteristics for each workload, splitting of dataset into train and test datasets, feature selection based on common values between Spearman, Pearson and Kendall tau methods, prediction of energy consumption values using the machine learning models of linear regression and random forest regression.

Evaluation of the success of this project will be the measure the accuracy of the machine learning model when predicting energy consumption of a workload based on its code characteristics, and the energy consumption reduction when applying suggested optimal settings to a given workload.

the following are the assessment criteria used to determine project success, stratified for each machine learning model:

1. Root Mean Square Error (RMSE) – standard deviation of predictions.
2. R^2 – coefficient of multiple determination for multiple regression, percentage variance between line of best fit and values.
3. Percentage variance average – average percentage difference between predicted values and actual values.

The project was concluded to be a success based on the results from Random Forest Regression model achieving an accuracy of 1.81% for energy consumption predictions and reducing energy consumption for given workloads by 22.55% through application of suggested optimal user configurable settings, thereby achieving the goal of this project to minimise energy consumption for HPC workloads.

Student ID: 201034888
Date: 02/10/2020

BACKGROUND RESEARCH:

For the initial conduct of this project a method to measure energy consumption and code characteristics for a workload on an HPC system was required, project testing was performed using the Barkla HPC system at the University of Liverpool.

Initial research for the project was centred around a standard method for the measurement of energy consumption, It is from this point that the PowerStack working group [6] was discovered due to their objective of creating a standard power management architecture for HPC systems, use of such a system for this project would have allowed for vendor-agnostic use of project software due to standard energy consumption recording API's, however after attending the PowerStack Seminar in June 2020 it was found that the PowerStack project is only in preliminary stages at present and a working solution was not available at the time of this project.

Another research topic considered was the Global Extensible Open Power Manager (GEOPM) framework [7] which allows for energy consumption measurements in a standard method across system and also performs energy optimisation by adjusting CPU frequencies and power, however due to the nature of this framework it was decided for the purposes of this project not to use such a framework as project results may have been unreliable as to whether energy efficiency gain was a result of this projects work or GEOPM optimisations.

Final energy consumption measurements and recording of code characteristics was performed using the Perf Tools for Linux application due to its low level access to performance counters and kernel events on Linux based systems, this tool allowed for access to energy consumption values, specifically using the Running Average Power Limit (RAPL) counters for energy consumption on Intel processor platforms to record energy consumption values for processor cores, entire processor die and CPU attached dedicated memory. [8]

This project was limited to CPU energy consumption (i.e. non-GPU workload energy consumption optimisation) therefore use of Core, Package and Memory energy consumption counters were appropriate when evaluating the energy consumption of a given workload, as the majority of system energy consumption is from CPU and memory energy consumption, with energy consumption of other components such as storage drives, fans, Motherboard components etc making up a minority of remaining system energy consumption.

Perf tools grants access to kernel and hardware level performance counters which allow for the recording of CPU operations for a given workload such as floating point operations, branching operations and integer arithmetic operations among others, through recording of these counters it was possible to determine the runtime characteristics of a workload thus allowing for the comparison of one workload to another based on these runtime characteristics.

Student ID: 201034888
Date: 02/10/2020

Due to the runtime of this project it was deemed unfeasible to record counters for each workload run on an HPC system however the software suite produced for this project allows for compatibility with such an implementation, instead a benchmarking suite was assembled using open-source benchmarking applications to simulate common workloads on an HPC system, simulated workloads allow for the prediction of energy consumption for real workloads based on shared code characteristics between benchmarks and real workloads, the benchmarking suite consists of the industry standard Linpack benchmark, Nasa Parallel benchmarks and Mantevo Mini-Apps to simulate HPC workloads.

Job submission and collection of performance counter records was done through a batch job scheduler, which required the submission of jobs using a Bash shell script, included in this project is a standardised example of a Bash shell script to run benchmarks and record performance counter results in a standard output format to enable compatibility with this projects software suite, an example of which can be found in appendix as "BATCHEXAMPLE.txt".

The example bash script for running a benchmark repeats execution in a loop iterating over all cores in an HPC system from single core execution to all available cores on a node, while recording performance counters for each execution, benchmarks are also stratified based on compiler used and optimisation level, the full list of counters collected using Perf Tools in this project along with their definition can be found in the appendix under "PERFeventcodes.txt".

Due to the predictive nature of this project, predicting an energy consumption value and seeking to minimise energy consumption of a workload, there was a significant machine learning component to this project from the outset as a result, this required research into machine learning methods such as models and prediction of a continuous value, feature selection from a dataset and creation of such datasets for the application of machine learning.

For the purposes of this project the Python programming language was used due its support for data science libraries and their application for machine learning in this project.

Principal Component Analysis was performed on the project dataset to reduce the number of features used in fitting a machine learning model, to both improve model performance by reducing execution time and also improve prediction accuracy through elimination of poorly correlated features, this process is known as Feature Selection.

Three methods from the SkLearn library were used for determining feature correlation to a continuous value, known as the target value for prediction which in the case of this project was total energy consumption, methods described in detail in design section:

1. Pearson's Correlation Coefficient
2. Spearman's Rank Coefficient
3. Kendall Rank Coefficient

Student ID: 201034888
Date: 02/10/2020

Use of the aforementioned feature correlation methods resulted in correlation values to target for each feature, the process of feature selection was to select the 10 most and least correlated values common amongst all 3 correlation methods results, where most and least correlated means: Most – As X increases Y increases, Least – As X increases Y decreases.

For the prediction of a value for total energy consumption the use of a machine learning model was required which could return a continuous value as output, as the target feature of this project is the measure of total energy consumption in joules.

Based on the input features and dataset given to a machine learning model, this was not a classification problem but a regression problem as such a regression machine learning model was required and two models were selected for use, details in design section:

1. Linear Regression
2. Random Forest Regression

Upon generation of a prediction for total energy consumption based on code characteristics in a training dataset, the following steps were devised for the generation of the suggested optimal user configurable settings, the stages are described below:

1. Using the predicted energy consumption value for a workload, compare predicted value to test dataset and find the nearest value which is an actual value in the test dataset.
2. Using the nearest test dataset value, determine the benchmark ID used to generate said value by fetching benchmark data for the nearest value row.
3. Using the fetched Benchmark ID, stratify training dataset for all rows matching fetched benchmark ID.
4. Select row from stratified training dataset with minimum total energy consumption value.
5. Fetch user variables from selected row, and return as suggested optimal settings for predicted energy consumption value.
6. These suggested optimal settings can then be used test if a reduction in energy consumption against predicted value is possible for a given workload.

Student ID: 201034888
Date: 02/10/2020

3. **DESIGN**:

The design stage of this project consisted of meeting the following objectives for data collection and system functionality:

1. Measurement of energy consumption and recording of performance data
2. Running Benchmarking suite with changing user variables
3. Collection of benchmark outputs
4. Creation of machine learning dataset
5. Prediction of energy consumption given dataset
6. Suggestion of optimal user variables for energy consumption reduction

The hypothesis tested in the project is defined as follows:

For a given code there exists the potential of optimisation in terms of energy efficiency (using the least amount of energy possible to get a solution), solely by knowing the characteristics of a given code, it is possible to predict its energy consumption and suggest user variable configuration to optimise for energy consumption reduction.

Prediction of energy consumption for given code is only possible through measurement of energy consumption for pre-existing code and recording of code characteristics through means of performance counters, such records can then be used for the prediction of energy consumption for other code based on similar characteristics.

- Recording of energy consumption for pre-existing code was performed using Perf tools and access to the Intel RAPL (Running Average Power Limit) utility, for recording of the following energy consumption values:

    1. Core
        i. Sum of energy consumption for all cores in a processor
        ii. Note: Core may not be available to Intel Processors released after Haswell Generation (2014)
    2. Package
        i. Energy consumption for entire processor die
        ii. Note: Standardised method for CPU energy consumption measurement after Haswell Generation (2015-Present)
    3. Memory
        i. Energy consumption for Random Access Memory attached to processor, also called Dedicated Memory

Student ID: 201034888
Date: 02/10/2020

The following were classified as benchmark characteristics for measuring defining features of different programs, the data points from which would be used for comparison to test programs and allow for energy consumption prediction through characteristic similarities to training benchmark data:

1. Integer Intensive
    - A workload that makes heavy use of integer instructions, all programs make use of integer instructions to some extent and so the energy efficiency with integer intensive workloads can indicate general energy efficiency and performance of a system.

2. Floating point intensive
    - A workload containing heavy use of floating-point operations typical processors contain integer execution units and floating-point execution units, thus the energy efficiency and performance of integer intensive operations may not correlate with the energy efficiency or performance of floating-point intensive operations for the same system.

3. Input / Output (I/O) Bound
    - A workload whose data structures cause a significant percentage of processor cycles to be stalled awaiting data to load, as a result of this limitation processor performance degrades, this measure is an indication of the input / output performance of a system, such workloads will benefit more from CPUs with high bandwidth cache memory and Random Access Memory, along with low latency and high bandwidth system storage. Due to the necessity of data being moved between CPU and system memory along a bus with a fixed data transfer rate, the bandwidth of which can limit the speed of computation, a limitation known as the Von Neumann bottleneck.

4. Memory bound
    - A workload whose total execution time is determined primarily by the capacity of memory available, latency of memory access and data transfer bandwidth. Such workloads benefit from greater memory capacity, bandwidth and low latency.

5. Compute bound
    - A workload whose total execution time is determined by the instructions per cycle and frequency rate of the CPU, an example workload containing many calculations on small amounts of data is likely to be compute bound and will benefit from an increase in instructions per cycle and frequency of CPU.

Use of these characteristics allows for the evaluation of different aspects of system performance in terms of Memory, CPU and I/O, as well as energy efficiency of different workloads, the assumption being that a CPU may have greater energy efficiency for certain workloads than others, note however that this classification list is a broad definition of measurements which were expanded upon during project realisation.

Student ID: 201034888
Date: 02/10/2020

The collection of energy consumption data and code characteristics were used with a Benchmarking suite containing well-known pre-existing HPC benchmarks which simulate common workloads. The benchmarking suite, notably the standardised batch submissions script was designed for use with the slurm workload manager as this was the job scheduler used for the test hpc system, however the benchmark suite can be used with other workload managers however this will require conversion of slurm commands to be compatible with other job scheduler systems.

All Benchmarking data was collated into a single dataset for further evaluation, the aim of the benchmarking suite is to create a standardised method for the measurement of code characteristics and energy consumption in an HPC system, with the result of standard output files which can then be converted to a dataset using the project software suite to allow for use of machine learning.

The Benchmarking suite created for this project consists of existing well-known HPC benchmarks which simulate common workloads. The suite is an amalgamation from three existing benchmarking suites, namely the NASA Parallel Benchmarks suite, HPC Challenge Linpack benchmark and Mantevo Mini-Apps benchmark suite, details of which are included below:

1. NASA Parallel Benchmarks
    - IS – Integer Sorting: Tests random memory access.
    - DC/DT – Data Cube: Unstructured computation, parallel input / output and data movement between CPU, Memory and Storage.

2. HPC Challenge Benchmark
    - Linpack: measure a system's floating-point performance through solving an X by Y system of linear equations.

3. Mantevo Mini-Apps Benchmarks
    - CloverLeaf – Hydrodynamics simulation using two-dimensional Eulerian formulation.
    - MiniAero – unstructured finite volume code that solves the compressible Navier-Stokes equations.
    - miniMD – A light-weight molecular dynamics application.

Student ID: 201034888
Date: 02/10/2020

The result of each benchmark execution in the suite was given a pre-determined name along with other information regarding the user variable configuration for each benchmark run as shown in the example below:

1. Benchmark ID
   - Number to identify row in dataset
2. Benchmark Name
   - Name of run benchmark
3. Optimisation level (Compiler)
   - Optimisation level of compiler used e.g. O0-O3
4. Platform
   - Manufacturer / Designer of CPU e.g. Intel, AMD, ARM
5. Architecture
   - Processor architecture e.g. x86, ARM, PowerPC
6. Processor name
   - Name of processor used for benchmark run
7. Processor generation
   - Name of processor generation
8. Threads used (Cores used)
   - Number of CPU cores used for benchmark run
9. Parallelisation Technology
   - How workload was parallelised e.g. OpenMP, MPI or Hybrid (OpenMP + MPI)

Standardised output files from the benchmarking suite are parsed for energy consumption and performance counter data, with all values added to a dataset for later use in machine learning by the software suite produced for this project, by evaluating data produced from benchmarking we observed correlations between code characteristics through performance counter data and energy consumption, where such a correlation exists the characteristics in question are selected for use in prediction of energy consumption, this project also planned to observe any differential in energy consumption for a given code across CPU architectures.

Standardised output files were achieved through the use of a standard format job submission batch script created for this project, which used a standardised method for the recording of user variable configurations, energy consumption measurement and performance counter measurement, such standard format batch script was modified for use with OpenMP and MPI parallelisation technologies, as well as use of benchmark specific required libraries to be loaded for a given benchmark run, an example of the standardised batch script can be found in the appendix of this project under the title: “BATCHEXAMPLE.txt”, as well as a used example of an MPI batch script under the title: “BATCHUSED.txt”.

Note however that both “BATCHEXAMPLE.txt” and “BATCHUSED.txt” are in the wrong format for sbatch submission in Slurm, when opened the user should configure the example batch scripts to their liking and save them with the “.sh” (shell) file extension to allow for usage with slurm’s sbatch system. Also, all benchmarks, output files and batch scripts can be found in the project appendix folder titled “eep” as this folder contains all results from testing.

Student ID: 201034888
Date: 02/10/2020

An exhaustive list of performance counters recorded through perf by the standardised batch script can be found in the appendix under the filename “Performance_counters_list.txt”. This list of performance counters is used as a measure of code characteristics which allow for energy consumption prediction through comparison to other code in machine learning.

Through these observations the goal was to perform data analysis on energy consumed over runtime for each benchmark code and architecture, then through use of machine learning over correlated features to generate a prediction for energy consumption for a program based on its features, and a suggestion of user variable configuration to optimise a program for minimal energy consumption, in the form of environment variables, job scheduler flags and CPU architecture for which the program should be executed.

To meet the aims of this project a Machine Learning software suite was created consisting of the following utilities:

- Slurm output parser
    - Reads benchmark output data and generates a comma separated values dataset.

- Dataset splitter
    - Divides a given dataset into training and testing datasets.

- Machine learning program
    - Requires training and testing datasets as input, perform feature selection and fitting of a machine learning model.
    - Makes a prediction using the testing dataset, also calculates an optimal user configuration of settings for a given workload.

Through the prediction of energy consumption for a given code, the suggestion of optimal user variable configuration is done though use of machine learning by selection of the closest datapoint to the energy consumption prediction value and gathering of said datapoints benchmark classification, performing dataset stratification based on benchmark identifier and selection of energy consumption minimum record in the strata, of which the user variable configuration will be returned to the user as a suggested energy consumption optimisation configuration for user variables, details of which can be found in realisation section.

Prior to the application of machine learning to a training dataset it is necessary to reduce the number of datapoints known hereon in as features, to be considered in order to improve machine learning model performance and accuracy by selecting features with greatest and least correlation to our target feature of total energy consumption.

Student ID: 201034888
Date: 02/10/2020

Features selected from the training dataset were chosen based on their correlation to total energy consumption. Each feature selection method used in this project calculates the correlation coefficient between each feature and our target feature of energy consumption, with a value between -1.0 and +1.0, +1 being a positive correlation (As X increases, Y increases) and -1 being a negative correlation (As X decreases, Y increases), correlation values were determined using three feature selection methods as detailed below:

1. Pearson's Correlation Coefficient
    - Measures only a linear correlation between 2 variables X and Y, this only applies for relationships between variables where the increase in value X is associated with a proportional change in variable Y, as Pearson's assumes normal distribution of values where proportional relationship does not change.

2. Spearman's Rank Coefficient
    - Measures non-linear correlation between 2 variables X and Y, through statistical dependence ranking of 2 variables it assesses whether the relationship between 2 values is monotonic, which is defined as whether the value of variable X increases, Y increases or as the value of X decreases, Y increases, however Spearman's differs from Pearson's as the rate of increase/decrease does not have to be constant, this is known as a monotonic relationship / function.

3. Kendall Rank Coefficient
    - Measures ordinal concordant and discordant association between variables X and Y, typically returns correlation values similar to Spearman's.

    *Note: Correlation is simply the degree to which two features are linearly related, high correlation = close to linear relationship, low correlation = non-linear relationship.

Machine learning for this project was designed around the prediction of a continuous value and therefore the selection of machine learning regression models was required.

Student ID: 201034888
Date: 02/10/2020

For the purposes of this project two regression models were selected for the creation of a prediction engine details of which are below:

1. Linear Regression
    - Machine learning model in which the target variable is a continuous value (a number not a classification), nature of the regression line is linear resulting in a “Line of best fit” through datapoints by establishing the relationship between the target variable (Y) and multiple independent variables (X).
    - This model can be represented by the equation Y=a+b*X + e, where a is the intercept, b is line degree and e is the error term, such equation can be used to predict the value of a target variable based on given independent predictor variable values, making for an obvious choice in machine learning where a predicted result is desired.

2. Random Forest Regression
    - Machine learning model with a non-linear nature, allowing for the creation of a prediction based on previously observed labels, as such the accuracy of random forest regression models increases proportional to dataset size, through the generation of many decision trees during model training this technique outputs the mean prediction of individual trees in the forest, which acts as a meta-estimator through aggregation of multiple decision trees to make predictions for a target variable (Y) based on predictor values (X), Random Forest Regression is typically a more accurate predictor model than linear regression however due to the nature of random forests (i.e. random decision trees) it can be difficult to know why the model made the predictions it did. However, owing to the possible accuracy of such a technique it was included in this project for the purposes of predicting energy consumption.

Application of either machine learning model to the projects training dataset results in the creation of a prediction engine which is then tested against the projects test dataset in which the selected model must predict the energy consumption (dependent variable) for each program in the test dataset using the code characteristics (independent variables) values, the test dataset contains an actual record of the programs energy consumption which allows for the comparison between the models predicted energy consumption value and the actual energy consumption value for a given code, this enables us to determine the accuracy of each machine learning model by calculating and recording the percentage difference between actual energy consumption values vs model predicted values.

Student ID: 201034888
Date: 02/10/2020

Upon generation of a prediction for total energy consumption based on code characteristics in a training dataset, the following steps were devised for the generation of the suggested optimal user configurable settings, the stages are described below:

1. Using the predicted energy consumption value for a workload, compare predicted value to test dataset and find the nearest value which is an actual value in the test dataset.
2. Using the nearest test dataset value, determine the benchmark ID used to generate said value by fetching benchmark data for the nearest value row.
3. Using the fetched Benchmark ID, stratify training dataset for all rows matching fetched benchmark ID.
4. Select row from stratified training dataset with minimum total energy consumption value.
5. Fetch user variables from selected row, and return as suggested optimal settings for predicted energy consumption value.
6. These suggested optimal settings can then be used test if a reduction in energy consumption against predicted value is possible for a given workload.

The final goal of the project was for a given codes characteristics, generate a prediction of energy consumption and suggest a user variable configuration to minimise energy consumption, note that the software suite produced for this project can only make predictions and suggest optimal configurations, implementation of suggested optimal configurations is dependent upon the system user following said suggestion when running a given workload, examples of energy consumption reduction as a result of following suggested user optimal configuration produced by this projects software suite can be found in the realisation and evaluation sections.

4. **REALISATION**:

The Projects design was implemented through the creation of two suites of software, a benchmarking suite and a Machine Learning Software Suite, which collected and transformed benchmark output data into a machine learning dataset, as well as performed feature selection, model training, energy consumption prediction and optimal user configuration suggestions. Complete details of these two software suites can be found in the design section.

Each program from the benchmark suite was written using the C Programming language, for each run of a benchmark the user variable configuration was changed in series to allow for the measurement of energy consumption and performance counters under different runtime environment conditions, this was necessary for the evaluation of energy consumption based on changing environment conditions such as compiler used and parallelisation technology, thus allowing for the suggestion of optimum user variable configurations for code with similar characteristics, this full list of variables was described and can be found in the design section.

Through collection of benchmark output files, which all have standardised datapoint outputs for our retrieval of user configuration, energy consumption and performance counter datapoints, it is required for this project collate all benchmark output files into a directory named “SlurmFiles” for use with the “Python-eepCSVGen” program in the Machine Learning software suite of this project.

Upon collection and parsing of all benchmark output datapoints, the “Python-eepCSVGen” produced a dataset consisting of 58 total feature columns and 4154 rows of data, this dataset is saved as “BenchmarkOut.csv” and was split 20% / 80% into testing and training datasets respectively by the “Python-EEPTrainTestSplit” program, creating “train.csv” and “test.csv” datasets as output, the train and test datasets were then used in the “Python-EEPML” program for training and testing of machine learning models for energy prediction and suggestion of optimal user variable configurations with regards to minimum energy consumption.

Student ID: 201034888
Date: 02/10/2020

The Machine Learning software suite consists of three distinct programs all of which perform separate functions and form a dependency chain with the outputs of one program required as inputs for the correct functioning of the next, the dependency chain operates as follows:

1. Python-eepCSVGen
    - Generates “BenchmarkOut.csv” dataset, containing collated datapoints from all benchmarks parsed during the execution of this program.
    - Requires input of benchmark output files into a directory named “SlurmFiles”, note: Output files are required to be in standardised format.

2. Python-EEPTrainTestSplit
    - Takes “BenchmarkOut.csv” generated by the above-mentioned program and splits it into training and testing datasets based on a percentage input by the user, generating “train.csv” and “test.csv” as output files. (e.g. Entering 20 will cause a 20% test / 80% train split, for the testing of this project a 20%/80% split was used for the generation of test.csv and train.csv files)

3. Python-EEPML
    - Takes “train.csv” and “test.csv” from above-mentioned program and performs feature selection using Pearson’s, Spearman’s and Kendall’s coefficients as described in the design section.
    - By using a mix of feature selection techniques we can find common features amongst them to select for use as important features for model fitting, as such the 10 most correlated features and 10 least correlated features as scored by the three feature selection techniques were used to create a feature set of 20 independent variables for machine learning model fitting to allow for the prediction of our 1 dependent variable of total energy consumption.
    - After feature selection process is finished the user is presented with the option to use one of the two machine learning models implemented in this project, Linear Regression or Random Forest Regression, a detailed description of both can be found in the design section.
    - Both machine learning models use the 20 selected features for prediction of total energy consumption, after training using the “train.csv” dataset, the selected model is applied to the “test.csv” dataset and using the 20 selected features generates a predicted value for total energy consumption, to evaluate the accuracy of this prediction in this projects testing an output file named “submission.csv” is generated which contains a dataset comprised of benchmark ID, actual energy consumption value, predicted energy consumption value and percentage variance.

    - Upon generation of predicted energy consumption values, the program then performs a nearest value search for each predicted energy consumption value to actual energy consumption values in the "train.csv" dataset, then stratifies the dataset based on benchmark and finds the local minimum for the associated benchmark, returning the user variable configuration used to the user as the suggested optimal user variable configuration for use with their evaluated code, details of this process are included in the design section.

The following Python libraries were used in the three programs which make up this project's software suite, this listing shows their name and description for reference, specific libraries used in each program are included in their section, they're required to be installed in Python environment for programs to run correctly.

- Libraries:
    - Pandas
        i. A data analysis and manipulation library used for the creation and manipulation of data frames.
    - NumPy
        i. Adds support for multi-dimensional arrays, matrices and high-level mathematical functions.
    - Matplotlib
        i. A plotting tool for python, providing an object-oriented API for the generation of graphs.
    - Seaborn
        i. A data visualisation tool providing a high-level interface for the creation of statistical graphics.
    - SkLearn
        i. A Machine Learning library featuring various feature selection, classification, regression and clustering algorithms including support vector machines.
    - OS
        i. Handles filenames, paths, directories and general file manipulation for any given Operating System that the python code is running in such as Linux, Mac OS or Windows.
    - RE
        i. Library enabling Regular expression engine use in Python, conforming to Perl Regular Expression engine naming conventions and format standards.

The total dataset size produced by "Python-eepCSVGen" was 4154 rows with 58 columns, where each row corresponds to a unique benchmark run and columns refers to all dataset features and datapoints collected from the benchmark output files.

Student ID: 201034888
Date: 02/10/2020

The resulting output files from benchmark runs must be transferred from system storage into the working directory of the python program Python-eepCSVGen, in a directory named “SlurmFiles”. The Python-eepCSVGen program uses the following libraries to perform its functions as described:

- Libraries:
    - Pandas
    - OS
    - RE

these output files are then parsed during program runtime using regular expression matching to retrieve data values which are then stored into a pandas dataframe containing all recorded datapoints such as energy consumption, user variable configuration and performance counters as described in the design section, note: output files must contain the “.out” file extension for use with this program, the produced dataframe is then output as a comma separated values file named “BenchmarkOut.csv” for use with Python-EEPTrainTestSplit.

The resulting “BenchmarkOut.csv” dataset file is input to the Python program Python-EEPTrainTestSplit which performs a randomised splitting of the input dataset into training and testing datasets for use in machine learning, by placing the generated “BenchmarkOut.csv” file into the “datasets” directory located in this programs working directory, this program will generate two datasets named “train.csv” and “test.csv” which will be split at a percentage ratio chosen by the user, however for the testing in this project all testing was done as a 20% / 80% split for testing and training respectively.

The Python-EEPTrainTestSplit program uses the following libraries to perform its functions as described:

- Libraries
    - OS
    - Pandas
    - SkLearn

The resulting “train.csv” and “test.csv” are intended for use with the Python program Python-EEPML and were placed in the “datasets” directory in that programs working directory for testing.

The resulting dataset sizes of “train.csv” and “test.csv” are as follows:

- train.csv
    - 3323 Rows, 59 Columns
- test.csv
    - 831 Rows, 59 Columns

Note: Columns refers to number of dataset features & rows refers to total dataset results.

Student ID: 201034888
Date: 02/10/2020

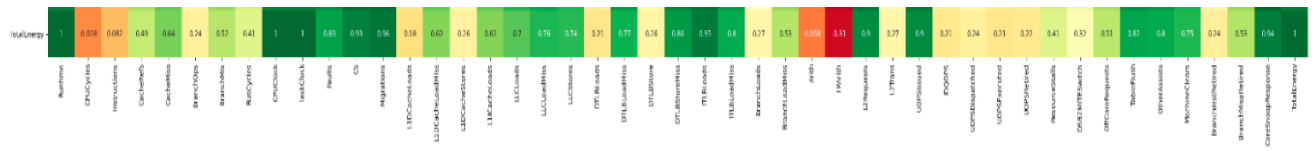
Resulting train and testing datasets are input to the program Python-EEPML which performs feature selection and model fitting to training dataset and application of trained model to test dataset in order to evaluate performance of the generated prediction model, this program also performs selection of suggested user variable configuration for the optimisation of energy efficiency for the execution of a given workload.

The Python-EEPML program uses the following libraries to perform its functions as described:

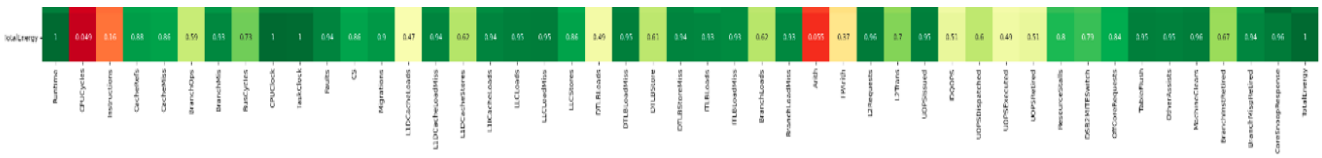
- Libraries
    - Pandas
    - NumPy
    - SkLearn
    - Matplotlib
    - Seaborn

The Feature Selection techniques used for this project were Pearson's Correlation Coefficient, Spearman's Rank Coefficient and Kendall Rank Coefficient, The labelled heatmaps below show the results of each feature selection technique which includes the score of correlation to the target variable for each independent variable, along with a heat map colouring of green = high correlation, red = low correlation.

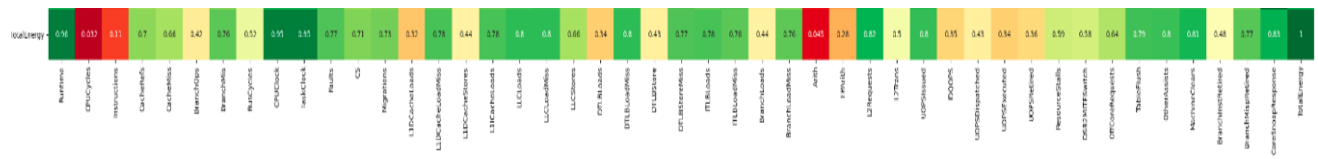
Pearson's Correlation Coefficient heatmap:



Spearman's Rank Coefficient heatmap:



Kendall Rank Coefficient heatmap:



As shown in all three heatmaps there exist features with high correlation scores and very low correlation scores, we exclude "TotalEnergy" from our considerations as this is our target feature.

Student ID: 201034888
Date: 02/10/2020

Using these three feature selection techniques we can perform principal component analysis to reduce the feature size our machine learning model must consider, to do this 20 features were selected based on correlation scores from the three feature selection techniques, 10 highest scoring features and 10 lowest scoring features, where correlation score can be defined as “green=high correlation = as X increases Y increases” and likewise “Red=low correlation = as X decreases Y increases” the resulting feature set is as follows:

- 20 considered features, ranked highest correlation to lowest
    - Runtime
    - CPUClock
    - TaskClock
    - Migrations
    - ITLBLoads
    - CS
    - UOPSIssued
    - L2Requests
    - DTLBStoreMiss
    - Faults
    - CPUCycles
    - Arith
    - Instructions
    - L1DCacheLoads
    - UOPSExecuted
    - DTLBLoads
    - IDQOPS
    - UOPSRetired
    - BranchInstRetired
    - UOPSDispatched

After feature selection is complete the 20 considered features can now be used by a machine learning model for training to create a prediction engine for the total energy consumption of a project given its characteristics based on the considered features we have selected.

The next stage in this program is the selection of a machine learning technique for model creation and fitting, the two machine learning models used in this project are Linear Regression and Random Forest Regression, detailed information about both can be found in the design section.

The User may choose the machine learning model of their preference through inputting 1 (Linear Regression) or 2 (Random Forest Regression) when prompted by the program, selecting either will begin the model fitting and training phase in which the machine learning model will use the dataset from “train.csv” for training and the dataset from “test.csv” for testing of the trained model, with a graphical representation of predicted energy consumption values vs actual energy consumption values displayed for user evaluation along with calculations for R^2 Variation and Root Mean Squared Error (RMSE).

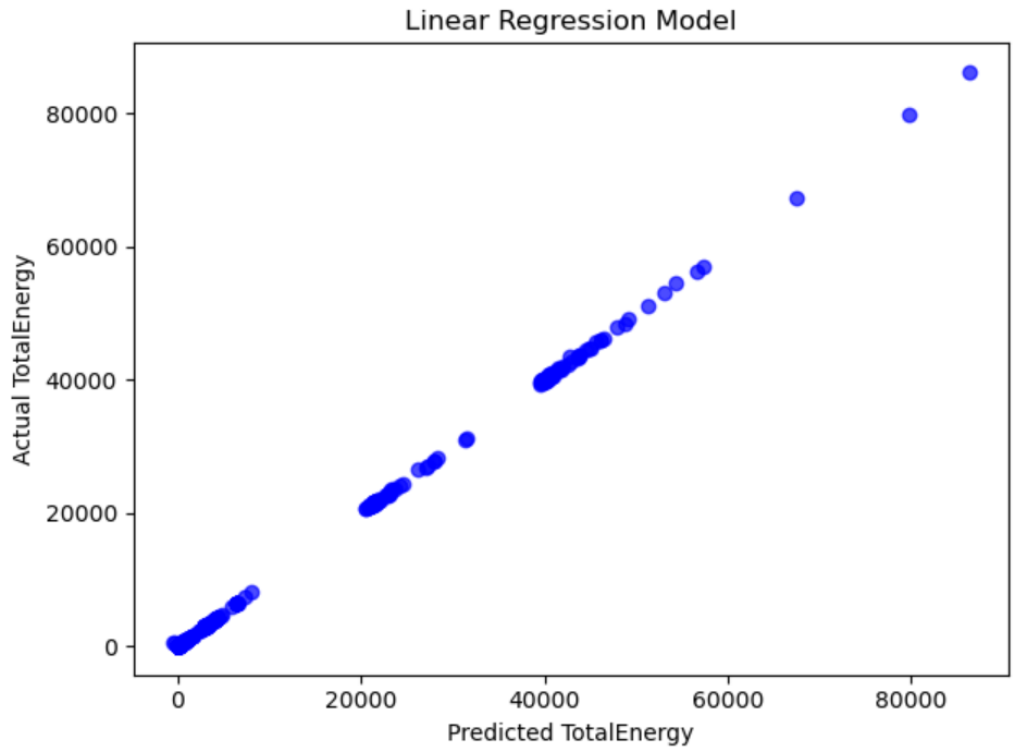
Student ID: 201034888
Date: 02/10/2020

Upon training of the machine learning model chosen, the model will perform energy consumption value predictions using “test.csv” dataset, generating an output file “submission.csv” which contains the Benchmark ID of each benchmark code considered, the actual energy total as measured by performance counters, the predicted energy total as calculated from selected features and the percentage difference between them.

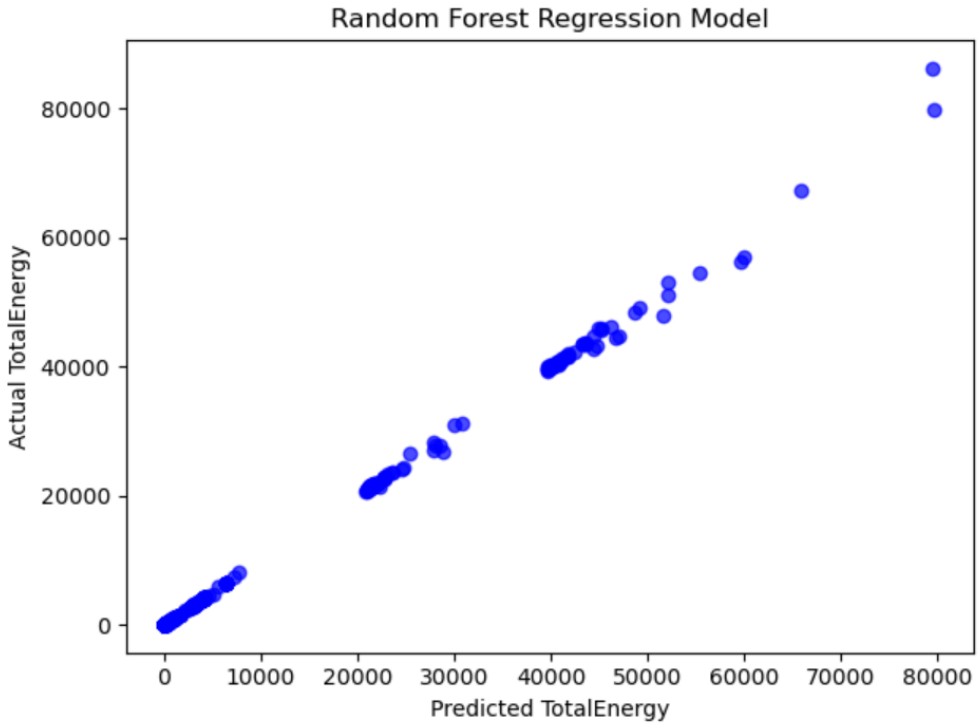
After calculation of predicted values the Python-EEPML program then calculates the suggested user variable configurations for optimising energy consumption, details of this process can be found in the design section, however this process returns the local minimum energy consumption user variable configuration for the nearest actual energy consumption value to the predicted value generated by the program, this is output as a dataset file named “PredictionOptimization.csv” which contains all the features and values from “submission.csv” but also includes the suggested optimal user variable configuration.

Included below are graphs showing the predicted vs actual values for total energy consumption for Linear Regression model and Random Forest Regression model respectively.

- Linear Regression



- Random Forest Regression



Through use of the suggested user variable configuration this project reduced energy consumption against the standard baseline, more information on this topic is discussed in the Conclusion.

5. **EVALUATION**:

It is my view that the project was an overall success in that all requirements and goals for this project were met with regards to the produced proof of concept using the Barkla HPC system.

The key points being the achievement of an average energy efficiency gain of 22.55% when using suggested optimal user settings compared to baseline settings for a test set of 6 benchmarks as shown in the Conclusion section.

The prediction of total energy consumption values for a program based on its code characteristics was also a success with an average percentage difference between actual energy consumption and predicted energy consumption of 1.81%, details of which can be found in the Conclusion section.

The current implementation does have some room for improvement with regards to the manual configuration of batch scripts when using different HPC systems, requirements for standardised output files. Another significant drawback to this project was a lack of third-party testing systems, as access to AMD and ARM HPC systems was achieved for this project.

However due to a lack of research time adequate testing could not be performed on such systems due to the necessity of proof of concept working on Barkla first, therefore a possible expansion of this project to HPC systems with different architectures would be an obvious choice.

Owing to development time constraints the implementation of machine learning models for selection is limited to two, this was intended to be expanded to four with the addition of support vector regression and neural network regression, as such if there was more development time these features could have been implemented.

Another topic for project expansion is the creation of a daemon tool for the recording of energy consumption and performance counters which can record the features of every code run on an HPC system, which would allow for a significant increase in dataset size rather than relying on benchmarks, also an expansion in the number of benchmarks is an obvious path to project expansion as greater diversity of code characteristics could yield greater prediction accuracy and possible energy savings from use of this projects software suite.

The software suite produced for this project is also transportable to any HPC system with a Python 3.8 environment installed and all necessary libraries, which should allow conducting research on other HPC systems require minimal reconfiguration.

6. **LEARNING POINTS**:

The key learning points with regards to knowledge acquired and skills learned during the completion of this project were a greatly expanded knowledge and skill with the Python programming language in general as well as the application of the language along with multiple libraries mentioned in this project document for the purposes of Machine Learning and data science, with regard to dataset manipulation and performance of regression techniques down to evaluating different models for accuracy and how dataset size influences application of machine learning models, as well as the practical use of principal component analysis in the form of feature selection including the application of multiple feature selection methods.

The low-level access to an HPC system in the form of hardware performance counters, energy consumption measurements and kernel event recording has given a skillset and knowledge into the inner workings of complex HPC systems with regards to energy consumption and optimising for energy efficiency.

Actions critical to the success of the project were the use of the Perf tools for Linux, which allowed for an all-in-one solution for the recording of energy consumption measurements as well as performance counters in a standardised format which allowed for the creation of an output file parser to transfer benchmark outputs into a usable csv dataset for machine learning.

The things I would do differently in this project would be the minimisation of setup time as roughly the month on June and 2 weeks of July were lost to HPC setup though liaising with Alces Flight and HPC services to get the installation of perf tools and proper user account level permissions to allow for low-level hardware access, by getting this process finished earlier or setup prior to project start would have meant greater development time available which would have allowed for the inclusion of more machine learning models to choose from such as support vector regression and neural network regression.

Another point of project expansion would be the increase in the number of benchmarks to expand the dataset size, a possible solution to dataset sparseness being the creation of a daemon tool which would automatically record code features for each code run on an HPC system.

7. **PROFESSIONAL ISSUES**:

This section concerns the projects relation to BCS code of conduct.

Public Interest Rules: As no human-supplied, derived or human participants took part in this project except for the author Public Interest rules 1 through 4 do not apply to this project and as such this project is in full compliance with public interest rules 1 through 4 inclusively.

Professional Competence and Integrity: Although this project did require the learning of new skills and knowledge as is inevitable with research projects, It is my judgement that such work was not outside my level of competence and I did not give any false representation of my abilities, nor did I accept any unethical inducement during my work or injure others, as such this project is in full compliance with Professional Competence and Integrity rules 1 thorough 7 inclusively.

Duty to Relevant Authority: This project work was conducted on behalf of myself whilst studying for an MSc in Advance Computer Science at the University of Liverpool and as such throughout this projects run and my time as a student I have observed all obligations and regulations which apply to me as a student, therefore this project is in compliance with Duty to Relevant Authority rules 1 through 5 inclusively.

Duty to the Profession: This project work was conducted by myself along with input from both Project Supervisors with regards to marking of previous work and goals in the project, at all times BCS regulations were adhered to and the standing of myself, colleagues, University of Liverpool nor BCS were ever brought into disrepute to the best of my knowledge, as such this project is in compliance with Duty to the Profession rules 1 through 6 inclusively in my judgement.

It is to the best of my judgement the conclusion that this project is in full compliance with all BCS code of conduct rules and regulations as described on their code of conduct website. [9]

Student ID: 201034888
Date: 02/10/2020

8. **CONCLUSIONS**:

The conclusion of the project yielded the following results, what follows is an example of the Linear Regression model vs Random Forest Regression model, prediction accuracy when compared to actual values and the suggested optimal user settings suggested when using each model, results for both models are included and labelled where necessary.

Below is an example of prediction accuracy with regards to predicting the energy consumption of a code based on its characteristics, this also includes a comparison of percentage variance between predicted energy values and actual energy values for each test benchmark ID, 20 results in total shown for each model here, the average percentage difference was calculated through the sum of all percentage variance shown below and division by number of rows (Calculated Mean).

Note however that the table figures shown below are not all results from testing, full details can be found in the "submission.csv" file included in the working directory of the "Python-EEPML" program, note however that the "submission.csv" file uploaded in the programs working directory contains results for Random Forest Regression model and not Linear Regression, to obtain live Linear Regression model figures the program must be run again and the Linear Regression model selected through input of "1" when prompted to choose machine learning model.

- Linear Regression Model: Energy Consumption Predictions

ID	ActualEnergy	PredictedEnergy	PercentageVariance
39	27.15	27.66	1.88
359	497.2	515.48	3.68
399	3099.58	3168.44	2.22
439	276.08	241.23	12.62
599	23337.55	23131.5	0.88
2013	438.5	328.87	25.0
1231	3294.36	3302.33	0.24
1728	2856.65	2913.99	2.01
3725	520.04	576.45	10.85
1736	21.48	24.0	11.73
1222	3717.4	3748.65	0.84
2367	1612.93	1597.87	0.93
96	12.22	14.62	19.64
764	22567.47	22679.46	0.5
109	19.58	24.0	22.57
3166	74.75	71.57	4.25
568	23672.73	23717.76	0.19
2446	9.86	16.09	63.18
2986	4077.86	4148.85	1.74
410	402.76	338.26	16.01

Student ID: 201034888
Date: 02/10/2020

- Linear Regression Model: Prediction based suggested optimisations

ID	Optimization	Platform	Architecture	Processor	Processor Gen	Compiler	Parallelization Tech	Threads Used
39	3	Intel	x86	Intel Xeon Gold 6138	Skylake	gcc	OpenMP	39
359	3	Intel	x86	Intel Xeon Gold 6138	Skylake	icc	Hybrid	38
399	3	Intel	x86	Intel Xeon Gold 6138	Skylake	icc	OpenMP	31
439	2	Intel	x86	Intel Xeon Gold 6138	Skylake	gcc	OpenMP	39
599	2	Intel	x86	Intel Xeon Gold 6138	Skylake	icc	OpenMP	21
2013	2	Intel	x86	Intel Xeon Gold 6138	Skylake	icc	OpenMP	40
1231	3	Intel	x86	Intel Xeon Gold 6138	Skylake	icc	OpenMP	31
1728	3	Intel	x86	Intel Xeon Gold 6138	Skylake	icc	OpenMP	31
3725	2	Intel	x86	Intel Xeon Gold 6138	Skylake	gcc	OpenMP	39
1736	0	Intel	x86	Intel Xeon Gold 6138	Skylake	gcc	OpenMP	4
1222	3	Intel	x86	Intel Xeon Gold 6138	Skylake	icc	OpenMP	31
2367	2	Intel	x86	Intel Xeon Gold 6138	Skylake	icc	OpenMP	40
96	3	Intel	x86	Intel Xeon Gold 6138	Skylake	gcc	OpenMP	33
764	2	Intel	x86	Intel Xeon Gold 6138	Skylake	icc	OpenMP	21
109	0	Intel	x86	Intel Xeon Gold 6138	Skylake	gcc	OpenMP	4
3166	2	Intel	x86	Intel Xeon Gold 6138	Skylake	gcc	OpenMP	39
568	2	Intel	x86	Intel Xeon Gold 6138	Skylake	icc	OpenMP	21
2446	0	Intel	x86	Intel Xeon Gold 6138	Skylake	gcc	OpenMP	4
2986	2	Intel	x86	Intel Xeon Gold 6138	Skylake	icc	OpenMP	40
410	2	Intel	x86	Intel Xeon Gold 6138	Skylake	icc	OpenMP	40

As the energy consumption table shows the average percentage difference for a predicted energy value vs an actual energy value is 10.048%, showing a fair degree of model accuracy for Linear Regression when predicting energy consumption values based on code characteristics. Other metrics used for model performance evaluation are R^2 Variance, RMSE and Percentage variance average, detailed descriptions of which are found in the design section.

Linear Regression model scores:

- R^2 Variance:
  - 0.9999458844247366

- RMSE distance between prediction and actual:
  - 8774.886291164772

- Percentage variance average:
  - 10.048%

Student ID: 201034888
Date: 02/10/2020

- Random Forest Regression Model: Energy Consumption Predictions

ID	ActualEnergy	PredictedEnergy	PercentageVariance
39	27.15	26.74	1.51
359	497.2	494.69	0.5
399	3099.58	3103.97	0.14
439	276.08	280.79	1.71
599	23337.55	23183.68	0.66
2013	438.5	429.44	2.07
1231	3294.36	3204.44	2.73
1728	2856.65	2895.45	1.36
3725	520.04	554.5	6.63
1736	21.48	20.97	2.37
1222	3717.4	3620.64	2.6
2367	1612.93	1607.53	0.33
96	12.22	11.85	3.03
764	22567.47	22538.93	0.13
109	19.58	19.53	0.26
3166	74.75	73.84	1.22
568	23672.73	23798.11	0.53
2446	9.86	9.82	0.41
2986	4077.86	4070.33	0.18
410	402.76	371.37	7.79

Student ID: 201034888
Date: 02/10/2020

- Random Forest Regression Model: Prediction based suggested optimisations

ID	Optimization	Platform	Architecture	Processor	Processor Gen	Compiler	Parallelization Tech	Threads Used
39	0	Intel	x86	Intel Xeon Gold 6138	Skylake	gcc	OpenMP	4
359	2	Intel	x86	Intel Xeon Gold 6138	Skylake	icc	OpenMP	40
399	3	Intel	x86	Intel Xeon Gold 6138	Skylake	icc	MPI	27
439	2	Intel	x86	Intel Xeon Gold 6138	Skylake	gcc	OpenMP	39
599	2	Intel	x86	Intel Xeon Gold 6138	Skylake	icc	OpenMP	21
2013	3	Intel	x86	Intel Xeon Gold 6138	Skylake	gcc	OpenMP	39
1231	3	Intel	x86	Intel Xeon Gold 6138	Skylake	icc	OpenMP	31
1728	3	Intel	x86	Intel Xeon Gold 6138	Skylake	icc	OpenMP	31
3725	2	Intel	x86	Intel Xeon Gold 6138	Skylake	icc	OpenMP	40
1736	3	Intel	x86	Intel Xeon Gold 6138	Skylake	gcc	OpenMP	33
1222	3	Intel	x86	Intel Xeon Gold 6138	Skylake	icc	OpenMP	31
2367	2	Intel	x86	Intel Xeon Gold 6138	Skylake	gcc	OpenMP	39
96	3	Intel	x86	Intel Xeon Gold 6138	Skylake	gcc	OpenMP	33
764	2	Intel	x86	Intel Xeon Gold 6138	Skylake	icc	OpenMP	21
109	2	Intel	x86	Intel Xeon Gold 6138	Skylake	icc	OpenMP	2
3166	2	Intel	x86	Intel Xeon Gold 6138	Skylake	gcc	OpenMP	39
568	2	Intel	x86	Intel Xeon Gold 6138	Skylake	icc	OpenMP	21
2446	2	Intel	x86	Intel Xeon Gold 6138	Skylake	gcc	OpenMP	36
2986	2	Intel	x86	Intel Xeon Gold 6138	Skylake	icc	OpenMP	40
410	2	Intel	x86	Intel Xeon Gold 6138	Skylake	icc	OpenMP	40

As the energy consumption table shows the average percentage difference for a predicted energy value vs an actual energy value is 1.808%, showing a high degree of model accuracy for Random Forest Regression when predicting energy consumption values based on code characteristics, which is an 82.006% improvement over Linear Regression model accuracy for the same 20 test benchmarks evaluated here. Other metrics used for model performance evaluation are R^2 Variance, RMSE and Percentage variance average, detailed descriptions of which are found in the design section.

Random Forest Regression model scores:

- R^2 Variance:
  - 0.9995579399288771
- RMSE distance between prediction and actual:
  - 71680.41435561625
- Percentage variance average:
  - 1.808%

Student ID: 201034888
Date: 02/10/2020

The testing performed in this project determined that Random Forest Regression returned a more accurate model for energy consumption predictions compared to linear regressions, therefore the following test was performed when applying the Random Forest Regression Model's Suggested optimal user variable configuration settings, we observed an average energy efficiency gain of 22.55% when using suggested optimal settings compared to baseline settings for a test set of 5 benchmarks as shown in the table below:

ID	Benchmark	Total energy consumption		Suggested optimal settings	Total energy consumption	Energy efficiency gain/loss
39	Mantevo Cloverleaf	27.15 Joules		GCC, O0, 4, OpenMP	11.43 Joules	57.90%
359	HPC Challenge Linpack	497.20 Joules		ICC, O2, 40, OpenMP	497.20 Joules	0.0%
399	Mantevo MiniAero	3099.58 Joules		ICC, O3, 27, MPI	3086.47 Joules	0.42%
439	Mantevo MiniMD	276.08 Joules		GCC, O2, 39, OpenMP	198.78 Joules	28.00%
599	NASAPB DC.A	23337.55 Joules		ICC, O2, 21, OpenMP	20550.26 Joules	11.94%
2013	NASAPB IS.C	438.50 Joules		GCC, O3, 39, OpenMP	276.07 Joules	37.04%

Baseline Settings:

Compiler: GCC	Optimisation level: O0	Cores Used: 40	Parallelisation technology: OpenMP

Another interesting thing to note from these results is the variation in compiler setting, optimisation settings and core / threads used count, as a naïve assumption would be to run all programs at O3 optimisation level, using the Intel compiler (if using Intel CPU), with maximum number of cores. With this assumption predicated on the idea that shorter runtime will mean lower energy consumption, however as these results show that is not the case and not only can use of this projects software suite minimise energy consumption for HPC workloads by an average of 22.55%, it also doesn't use maximum computational resources in terms of core / thread count to do it, thereby reducing both computational resource usage and total energy consumption for CPU and Memory.

- Average energy efficiency percentage change using suggested optimal user variable configuration:
    - 22.55%

The conclusion of this research paper is that energy efficiency gains of 20%+ are possible through use of optimal settings suggested by this projects machine learning software and thanks to the portable nature of Python code the software suite for this project could be run in any Python 3.8 environment which includes the required libraries allowing for use on any HPC system. However as mentioned in the evaluation section this project could be easily expanded upon through creation of a daemon tool to record performance counters for all workloads run on an HPC system.

9. **BIBLIOGRAPHY**:

[1] – Barkla HPC system, Link: https://www.liverpool.ac.uk/csd/advanced-research-computing/facilities/high-performance-computing/ [accessed 02/10/2020]

[2] – Perf tools, Link: https://perf.wiki.kernel.org/index.php/Main_Page [accessed 03/07/2020]

[3] – NASA Parallel Benchmarks suite, Link https://www.nas.nasa.gov/publications/npb.html [accessed 06/07/2020]

[4] – HPC Challenge Linpack benchmark, Link https://icl.utk.edu/hpcc/ [accessed 06/07/2020]

[5] – Mantevo Mini-Apps benchmark suite, Link https://mantevo.github.io/ [accessed 06/07/2020]

[6] – Powerstack API working group, Link https://powerstack.caps.in.tum.de [accessed 25/06/2020]

[7] – GEOPM, Link https://geopm.github.io [accessed 25/06/2020]

[8] – Intel RAPL, Link http://web.eece.maine.edu/~vweaver/projects/rapl/ [accessed 06/07/2020]

[9] – BCS Code of Conduct, Link: https://www.bcs.org/membership/become-a-member/bcs-code-of-conduct/ [accessed 02/10/2020]

10. **APPENDICES**:

Files included in Appendix are:

1. Folder - “eep” – contains full Barkla workspace and results used for project testing

2. Folder – “Python-Programs” – Contains software suite from project
   - Folder – “Python-eepCSVGen” – CSV Generator program file directory
   - Folder – “Python-EEPTrainTestSplit” – CSV Train/Test splitter program file directory
   - Folder – “Python-EEPML” – Machine Learning program file directory

3. File – “Slurmexample.out” – example of Slurm output file used for benchmark test

4. File – “PERFeventCodes.txt” – Contains perf event codes and hardware register values for Barkla HPC

5. File – BATCHEXAMPLE.txt – example batch script

6. File – BATCHUSED.txt – example of batch script used for benchmark test

7. File – “InputTestVals.csv” – Contains 6 rows used for baseline testing as shown in conclusion section

8. File – “COMP702 – User Guide.pdf” – Contains project user guide for how to use all programs in software suite and submit a job using example batch script.

9. File – “Performance_counters_list.txt” – Contains an exhaustive list of all performance counters used in Slurm batch script for this project.