***Winter Internship Project Report***

***Name: Aditya Periwal***

***Id: 201712018***

***Project: Performance Prediction using Machine Learning***

***Mentor: Prof. Amit Mankodi***

***Index***

|  |  |
| --- | --- |
| Sno. | Content |
| 1. | Introduction |
| 2. | Context Diagram |
| 3. | Scope statement |
| 4. | Tools and Libraries used |
| 5. | Testing Strategies |
| 6. | Lessons Learnt |

**Introduction**

There is a lot of research going on to predict the runtime of an algorithm. And quite remarkably, through Machine learning, it is possible to train models capable of predicting the runtime based on different hardware features.

The runtime depends on various features like BIOS and the Operating system, but most importantly it depends on the hardware features that are actually used to perform the execution of the instructions, so we can create a good model to predict runtime quite accurately using the hardware features alone.

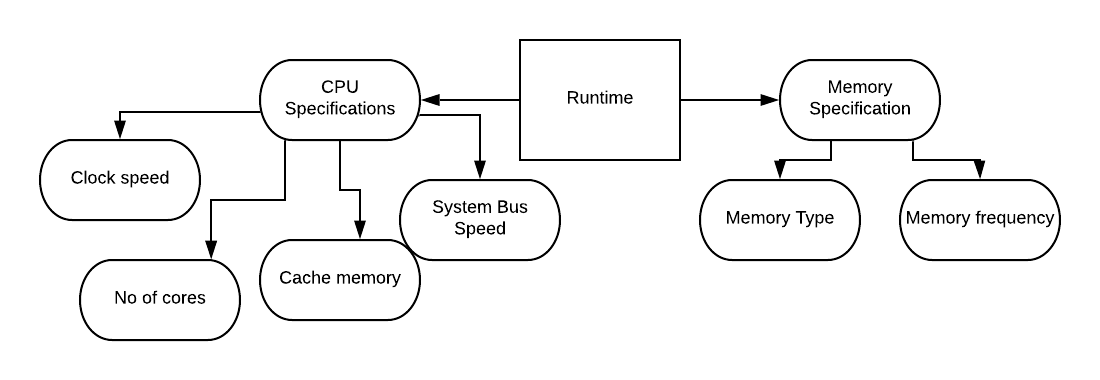


Fig1: Important Components that affects Runtime of an algorithm

The project was done as part of our winter internship and it aimed at creating a model that can predict the runtime of an algorithm. The runtime changes based on different system configurations, as the hardware features are different. So, we created a machine learning model capable of predicting the runtime of an algorithm based on the different hardware features alone.

The benchmark information used for the model was collected from SPEC CINT2006 Results for integer algorithms and SPEC CFP2006 Results for floating point algorithms, which are a part of Standard Performance Evaluation Corporation (SPEC) ([spec.org](https://www.spec.org)) benchmarks. The Benchmarks contained user submitted information regarding runtimes for different hardware configurations for different algorithms.

The Prediction model was trained on different hardware features such as clock-speed, no of cores, no of threads, memory-type, memory-frequency, cache-associativity, cache-memory and bus speed.

A good reference to know more about hardware features is: [sample processor information](https://ark.intel.com/content/www/us/en/ark/products/46472/intel-core-i3-530-processor-4m-cache-2-93-ghz.html) and [additional hardware information](http://www.cpu-world.com/CPUs/Pentium_4/Intel-Pentium%204%20670%203.8%20GHz%20-%20JM80547PG1122MM%20-%20HH80547PG1122MM%20(BX80547PG3800F).html).

The benchmarks were good but they lacked some important information to create an accurate model. So, we also gathered some additional information like the associativity and bus speed from websites like CPU-world ([CPU-world](http://www.cpu-world.com)).

The project initiated with collecting the benchmark information for different algorithms for each of the Integer and Floating Point types. The collected benchmarks were then extracted to create dataset files with around 1600 different configurations for each algorithm to be used for machine learning.

The Machine learning model was then trained using supervised learning with the dataset files. The learned model was tested using the different train-test splits for the datasets and the predictions were compared with the actual runtimes to calculate the accuracy of the model. The Project was successful in creating a model capable of predicting the runtime quite accurately (with around 2-3% error) for most of the algorithms it learned.

**Context Diagram**

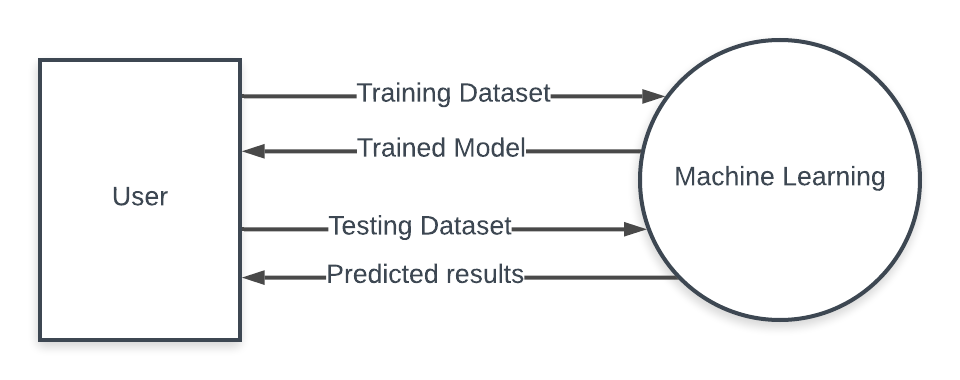


Fig2: Context Diagram

**Scope Statement:**

Scope Description

The project will collect the required benchmarks for all the algorithms that are to be used for the prediction and should be able to create a machine learning model that can predict the runtime of the algorithms when provided with the correct hardware specifications of the underlying system. The Project will also include analysis of the prediction done by the machine learning model.

Deliverables:

* The Project should be able to create datasets for the concerned algorithms for the prediction purpose.
* The Project should be able to train a model based on the Dataset created from the benchmarks.
* The Project should deliver a learned machine learning model that can predict the runtime of algorithms when provided with the correct hardware specifications from the dataset.
* The Project should include analysis of the results fetched by the learned machine learning model for different splits, for different algorithms.

Acceptance Criteria:

* The Dataset should reflect the correct values provided in the benchmarks for all the algorithms.
* The model should be able to predict the runtime for all the specifications for the same algorithm.
* The highest mean error in prediction of runtime done by the learned model should not be more than 10%.
* The highest median error in the prediction of runtime must not be more than 4%.

Constraints:

* Limited information regarding hardware features: As we did not had all the data, we worked with the limited amount of data available publicly by organizations like SPEC.
* Missing values for some features in the benchmarks: Even in the benchmark provided by SPEC some of the important fields had missing values for some entries. So we had no choice but to remove those entries with missing information from our Dataset.
* Time constraint for gathering information for benchmarks: As SPEC data was limited, we searched for the missing information on websites like CPU world which had a scraping limit of around 150 page requests for a particular day.
* Lack of other information like BIOS, OS and processor architecture in the benchmarks: The benchmarks provided by the SEPC don’t provide other useful information like BIOS flags, OS and processor architecture.

**Tools, Technologies, and Libraries used:**

Tools used:

* Jupyter Notebook: The Jupyter Notebook is an open-source web application that lets you to make and share documents that encompass live code, calculations, conceptions and descriptive text.
* Visual Studio code: Visual Studio Code is a source-code editor developed by Microsoft for Windows, Linux and macOS. It includes support for debugging, entrenched Git control, syntax highlighting, intelligent code completion, scraps, and code refactoring.

Programming Language(s) used:

* Python: Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991. It provides constructs that enable clear programming on both small and large scales.

Libraries used:

* BeautifulSoup: A library for web scraping. We used this in order to scrape for the required information regarding hardware from Spec benchmarks and CPU world.
* Requests: A standard library for making HTTP requests in Python.
* Urllib: A library for handling URL operations in Python.
* Csv: A library to work with Csv files. Most of our operations were CSV based so we did the csv operations using this library.
* Regex: A library for using regular expressions. We used it mostly to extract required information out of the collected benchmarks.
* Selenium: A library to use Selenium tool in Python. We used Selenium for automating the manual searching process for the required information.
* Numpy: A library to support large arrays and matrices in Python.
* Scikit-learn: A machine learning library for Python. We used it extensively for the machine learning part and for analysis.
* Pandas:A library for data manipulation and analysis in Python. We used it to handle csv using data-frames so that the manipulation of the data becomes relatively easier.
* Matplotlib: A 2d plotting library for Python. We used to plot the results into various formats like bar graphs, scatter diagrams and box plots.

**Testing Strategies and Reports:-**

Testing strategy:

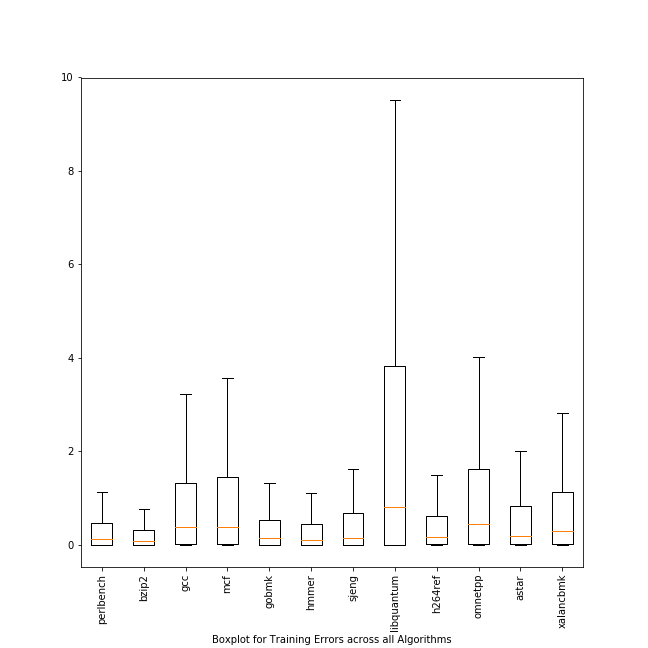
The model was tested using a method known as train-test split. The Dataset was spliced into train and test set in various ratios multiple times. The ratios used were 50-50, 60-40, 70-30 and 80-20. The model was trained using the train set and was then tested for prediction accuracy using the spliced test set. This way we were able to see if the Dataset is enough for proper learning of the model.

For instance for the split 80-20 the total Dataset was divided into two parts in which 80% of the architectures went into training set and rest 20% were used for the testing purpose to know the prediction accuracy of the model.

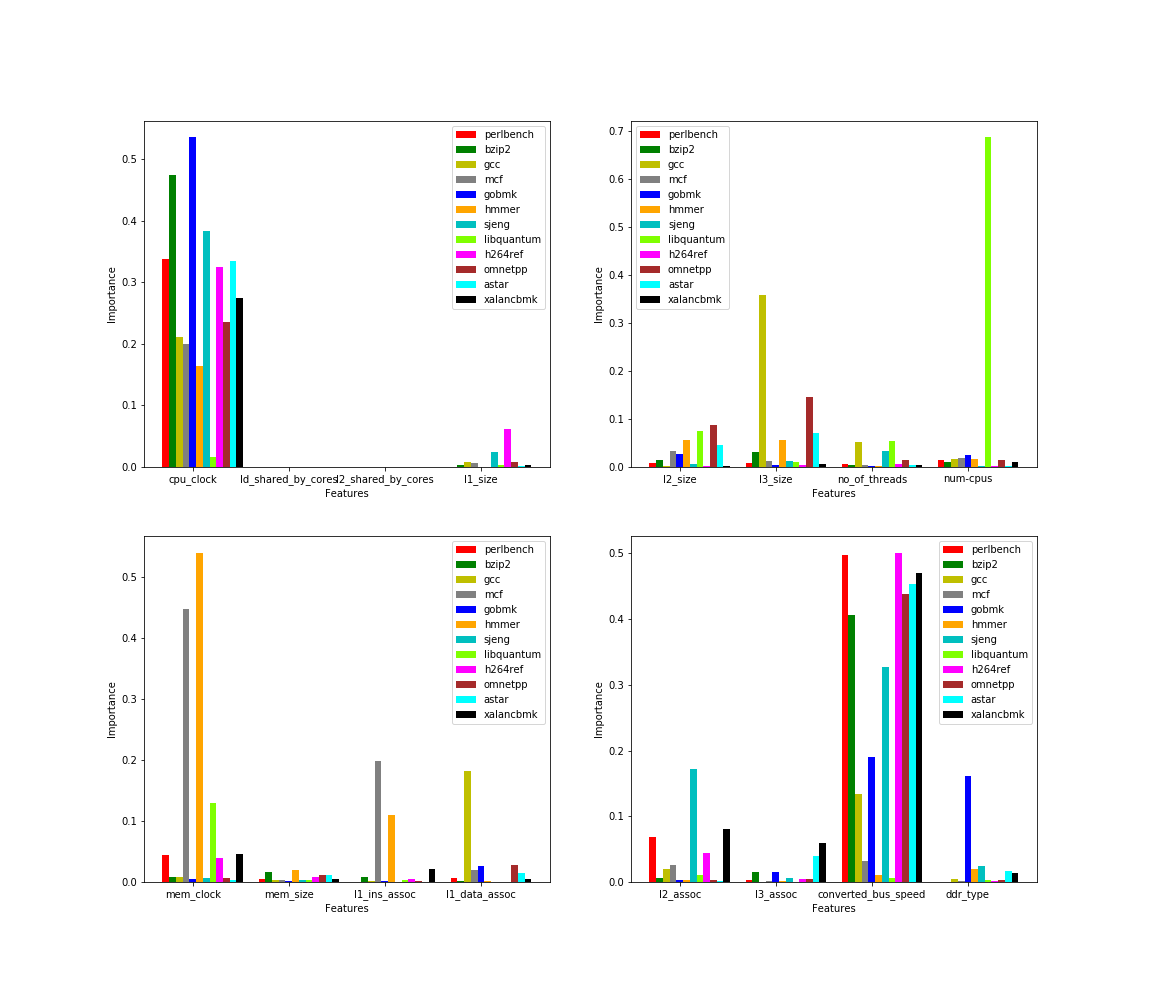
Reports:

The Following reports were generated after the completion of the project:

1. Box-plot depicting median training error in all algorithms for Integer type for the split 70-30:



1. Feature Importance graph for all the algorithms(split independent):



**Lessons Learnt**

The project taught me the following things:

* Important of team work: The project taught me how important it is to work with team and that too with proper communication regarding the progress of the project.
* Importance of writing dynamic code: The project taught me the importance of writing dynamic code so that it’s generic and can be used for similar work without changing much of the code.
* Importance of managing file versions: Initially we were lacking proper version control for the project, but soon we faced a lot of problems like lack of proper interfacing between different pieces of code and we realized how important it is to manage the versioning of the files used in the project.