UNIVERSITY OF OSLO

MASTER'S THESIS

Black-box performance modelling and analysis of microservice systems

Author:
Magnus NordbøMagnus NORDBØ

Supervisor:
Hui SongHui SONG

A thesis submitted in fulfillment of the requirements for the degree of Master of Informatics

in the

Faculty of Mathematics and Natural Sciences
Department Of Informatics

May 8, 2023

Declaration of Authorship

I, Magnus NORDBØ, declare that this thesis titled, "Black-box performance modelling and analysis of microservice systems" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

"Thanks to my solid academic training, today I can write hundreds of words on virtually any topic without possessing a shred of information, which is how I got a good job in journalism."

Dave Barry

UNIVERSITY OF OSLO

Abstract

Faculty of Mathematics and Natural Sciences
Department Of Informatics

Master of Informatics

Black-box performance modelling and analysis of microservice systems

by Magnus NORDBØ

The Thesis Abstract is written here (and usually kept to just this page). The page is kept centered vertically so can expand into the blank space above the title too...

Acknowledgements

The acknowledgments and the people to thank go here, don't forget to include your project advisor...

Contents

List of Figures

List of Tables

List of Abbreviations

MTS	**M**ultivariate **T**ime **S**eries
DTW	**D**ynamic **T**ime **W**arping

For/Dedicated to/To my…

Chapter 1

Introduction

1.1 Keywords

Observability
Microservices
Signal-to-noise ration

1.2 Abstract

This thesis project aims to explore the practicality and efficacy of multivariate time series classification to quickly diagnose performance bottlenecks in a microservice system. A simple microservice system running in Docker was forked from another project and minimally adjusted. Metrics were collected about the system using the Prometheus monitoring software, and stress testing was done at the API level with Pumba. The resulting data was fed into different MTS machine learning algorithms to judge their accuracy in predicting the source of the stress. Finally, the results of these predictions were compared and the methods for collecting, treating and training on the data were documented to benefit future implementations of a similar approach.

1.3 Research area and questions

The research area for this thesis is microservices and machine learning. An inevitable intersection appears between the two fields when the need for analysis of data from microservice systems arises. This is because these complex systems are capable of generating a vast amount of metric data about themselves. The amount of data is too large for a human to read, comprehend and reason about in any efficient capacity. This can make it very difficult to draw useful conclusions about the system. This is where machine learning comes in as a natural answer to the problem: While humans are limited in the amount of information they can comprehend, machines typically only benefit from absurdly large datasets to draw from, as long as the data is of acceptable quality. However, the field of machine learning applications on microservice system data is still immature as of time of writing. This thesis seeks to analyze a small subset of this field. The scope will be contained to identifying latency issues resulting from disproportionally high load in a part of the system, using only data from centralized logging.
Research question 1: How good are current machine learning classification algorithms at identifying causes for anomalous behavior in microservice systems?

Research question 2: What are the current main challenges to effectively utilize metric data from microservice systems?

1.4 Approach

To research these questions and try to create meaningful results that can have a real impact, I formulated a hypothesis to work as the base for the project. ***It is possible to use stress testing software and centralized logging to create a machine learning model of the system behavior to classify and identify sources of latency in a microservice system.*** To test this hypothesis, a fork of the open source demo microservice project "sockshop" by Weaveworks was used **Microservices-demo-sockshop**. This system was hosted on a local machine and stress tested using various configurations. Locust, an API stress testing tool. The full spectrum of available Prometheus metric data was collected in 601 time series features into csv files and labeled. There were a total of five classes, representing different endpoints connecting to different underlying microservices. This data was then analyzed both manually and mathematically, and preprocessed to reduce noise and extract important features. Several methods of preprocessing and classifying were quantitatively compared. Finally, the results of the analysis were discussed and conclusions drawn.

1.5 List of terms

- **Dynamic Time Warping**: A method of statistically comparing time series across time points to judge similarity.
- **Multivariate Time Series (MTS)**: Time series data with two or more variables.
- **Instance**: All the data collected from a system in a specific time frame, represented as an MTS.
- **Feature**: A unique Prometheus metric that expresses some kind of information about the system as a number. Collected at fixed intervals to form an MTS.
- **Time frame**: The period of time recorded in a time series. Each instance in this thesis corresponds to a specific time frame.
- **Centralized logging**: Umbrella term for various microservice info logging methods that collect logs from individual microservices and stores them together.
- **Distributed tracing**:

1.6 Learning LaTeX

LaTeX is not a WYSIWYG (What You See is What You Get) program, unlike word processors such as Microsoft Word or Apple's Pages. Instead, a document written for LaTeX is actually a simple, plain text file that contains *no formatting*. You tell LaTeX how you want the formatting in the finished document by writing in simple commands amongst the text, for example, if I want to use *italic text for emphasis*, I write the `\emph{text}` command and put the text I want in italics in between the curly braces. This means that LaTeX is a "mark-up" language, very much like HTML.

1.6.1 A (not so short) Introduction to LaTeX

If you are new to LaTeX, there is a very good eBook – freely available online as a PDF file – called, "The Not So Short Introduction to LaTeX". The book's title is typically shortened to just *lshort*. You can download the latest version (as it is occasionally updated) from here: http://www.ctan.org/tex-archive/info/lshort/english/lshort.pdf

It is also available in several other languages. Find yours from the list on this page: http://www.ctan.org/tex-archive/info/lshort/

It is recommended to take a little time out to learn how to use LaTeX by creating several, small 'test' documents, or having a close look at several templates on: http://www.LaTeXTemplates.com
Making the effort now means you're not stuck learning the system when what you *really* need to be doing is writing your thesis.

1.6.2 A Short Math Guide for LaTeX

If you are writing a technical or mathematical thesis, then you may want to read the document by the AMS (American Mathematical Society) called, "A Short Math Guide for LaTeX". It can be found online here: http://www.ams.org/tex/amslatex.html under the "Additional Documentation" section towards the bottom of the page.

1.6.3 Common LaTeX Math Symbols

There are a multitude of mathematical symbols available for LaTeX and it would take a great effort to learn the commands for them all. The most common ones you are likely to use are shown on this page: http://www.sunilpatel.co.uk/latex-type/latex-math-symbols/

You can use this page as a reference or crib sheet, the symbols are rendered as large, high quality images so you can quickly find the LaTeX command for the symbol you need.

1.6.4 LaTeX on a Mac

The LaTeX distribution is available for many systems including Windows, Linux and Mac OS X. The package for OS X is called MacTeX and it contains all the applications you need – bundled together and pre-customized – for a fully working LaTeX environment and work flow.

MacTeX includes a custom dedicated LaTeX editor called TeXShop for writing your '`.tex`' files and BibDesk: a program to manage your references and create your bibliography section just as easily as managing songs and creating playlists in iTunes.

1.7 Getting Started with this Template

If you are familiar with LaTeX, then you should explore the directory structure of the template and then proceed to place your own information into the *THESIS INFORMATION* block of the `main.tex` file. You can then modify the rest of this file to your unique specifications based on your degree/university. Section **??** on page **??** will help you do this. Make sure you also read section **??** about thesis conventions to get the most out of this template.

If you are new to LaTeX it is recommended that you carry on reading through the rest of the information in this document.

Before you begin using this template you should ensure that its style complies with the thesis style guidelines imposed by your institution. In most cases this template style and layout will be suitable. If it is not, it may only require a small change to bring the template in line with your institution's recommendations. These modifications will need to be done on the `MastersDoctoralThesis.cls` file.

1.7.1 About this Template

This LaTeX Thesis Template is originally based and created around a LaTeX style file created by Steve R. Gunn from the University of Southampton (UK), department of Electronics and Computer Science. You can find his original thesis style file at his site, here: `http://www.ecs.soton.ac.uk/~srg/softwaretools/document/templates/`

Steve's `ecsthesis.cls` was then taken by Sunil Patel who modified it by creating a skeleton framework and folder structure to place the thesis files in. The resulting template can be found on Sunil's site here: `http://www.sunilpatel.co.uk/thesis-template`

Sunil's template was made available through `http://www.LaTeXTemplates.com` where it was modified many times based on user requests and questions. Version 2.0 and onwards of this template represents a major modification to Sunil's template and is, in fact, hardly recognisable. The work to make version 2.0 possible was carried out by Vel and Johannes Böttcher.

1.8 What this Template Includes

1.8.1 Folders

This template comes as a single zip file that expands out to several files and folders. The folder names are mostly self-explanatory:

Appendices – this is the folder where you put the appendices. Each appendix should go into its own separate `.tex` file. An example and template are included in the directory.

Chapters – this is the folder where you put the thesis chapters. A thesis usually has about six chapters, though there is no hard rule on this. Each chapter should go in its own separate `.tex` file and they can be split as:

- Chapter 1: Introduction to the thesis topic
- Chapter 2: Background information and theory
- Chapter 3: (Laboratory) experimental setup
- Chapter 4: Details of experiment 1
- Chapter 5: Details of experiment 2
- Chapter 6: Discussion of the experimental results
- Chapter 7: Conclusion and future directions

This chapter layout is specialised for the experimental sciences, your discipline may be different.

Figures – this folder contains all figures for the thesis. These are the final images that will go into the thesis document.

1.8.2 Files

Included are also several files, most of them are plain text and you can see their contents in a text editor. After initial compilation, you will see that more auxiliary files are created by LaTeX or BibTeX and which you don't need to delete or worry about:

example.bib – this is an important file that contains all the bibliographic information and references that you will be citing in the thesis for use with BibTeX. You can write it manually, but there are reference manager programs available that will create and manage it for you. Bibliographies in LaTeX are a large subject and you may need to read about BibTeX before starting with this. Many modern reference managers will allow you to export your references in BibTeX format which greatly eases the amount of work you have to do.

MastersDoctoralThesis.cls – this is an important file. It is the class file that tells LaTeX how to format the thesis.

main.pdf – this is your beautifully typeset thesis (in the PDF file format) created by LaTeX. It is supplied in the PDF with the template and after you compile the template you should get an identical version.

main.tex – this is an important file. This is the file that you tell LaTeX to compile to produce your thesis as a PDF file. It contains the framework and constructs that tell LaTeX how to layout the thesis. It is heavily commented so you can read exactly what each line of code does and why it is there. After you put your own information into the *THESIS INFORMATION* block – you have now started your thesis!

Files that are *not* included, but are created by LaTeX as auxiliary files include:

main.aux – this is an auxiliary file generated by LaTeX, if it is deleted LaTeX simply regenerates it when you run the main `.tex` file.

main.bbl – this is an auxiliary file generated by BibTeX, if it is deleted, BibTeX simply regenerates it when you run the `main.aux` file. Whereas the `.bib` file contains all the references you have, this `.bbl` file contains the references you have actually cited in the thesis and is used to build the bibliography section of the thesis.

main.blg – this is an auxiliary file generated by BibTeX, if it is deleted BibTeX simply regenerates it when you run the main `.aux` file.

main.lof – this is an auxiliary file generated by LaTeX, if it is deleted LaTeX simply regenerates it when you run the main `.tex` file. It tells LaTeX how to build the *List of Figures* section.

main.log – this is an auxiliary file generated by LaTeX, if it is deleted LaTeX simply regenerates it when you run the main `.tex` file. It contains messages from LaTeX, if you receive errors and warnings from LaTeX, they will be in this `.log` file.

main.lot – this is an auxiliary file generated by LaTeX, if it is deleted LaTeX simply regenerates it when you run the main `.tex` file. It tells LaTeX how to build the *List of Tables* section.

main.out – this is an auxiliary file generated by LaTeX, if it is deleted LaTeX simply regenerates it when you run the main `.tex` file.

So from this long list, only the files with the `.bib`, `.cls` and `.tex` extensions are the most important ones. The other auxiliary files can be ignored or deleted as LaTeX and BibTeX will regenerate them.

1.9 Filling in Your Information in the `main.tex` File

You will need to personalise the thesis template and make it your own by filling in your own information. This is done by editing the `main.tex` file in a text editor or

your favourite LaTeX environment.

Open the file and scroll down to the third large block titled *THESIS INFORMATION* where you can see the entries for *University Name*, *Department Name*, etc ...

Fill out the information about yourself, your group and institution. You can also insert web links, if you do, make sure you use the full URL, including the `http://` for this. If you don't want these to be linked, simply remove the `\href{url}{name}` and only leave the name.

When you have done this, save the file and recompile `main.tex`. All the information you filled in should now be in the PDF, complete with web links. You can now begin your thesis proper!

1.10 The `main.tex` File Explained

The `main.tex` file contains the structure of the thesis. There are plenty of written comments that explain what pages, sections and formatting the LaTeX code is creating. Each major document element is divided into commented blocks with titles in all capitals to make it obvious what the following bit of code is doing. Initially there seems to be a lot of LaTeX code, but this is all formatting, and it has all been taken care of so you don't have to do it.

Begin by checking that your information on the title page is correct. For the thesis declaration, your institution may insist on something different than the text given. If this is the case, just replace what you see with what is required in the *DECLARATION PAGE* block.

Then comes a page which contains a funny quote. You can put your own, or quote your favourite scientist, author, person, and so on. Make sure to put the name of the person who you took the quote from.

Following this is the abstract page which summarises your work in a condensed way and can almost be used as a standalone document to describe what you have done. The text you write will cause the heading to move up so don't worry about running out of space.

Next come the acknowledgements. On this page, write about all the people who you wish to thank (not forgetting parents, partners and your advisor/supervisor).

The contents pages, list of figures and tables are all taken care of for you and do not need to be manually created or edited. The next set of pages are more likely to be optional and can be deleted since they are for a more technical thesis: insert a list of abbreviations you have used in the thesis, then a list of the physical constants and numbers you refer to and finally, a list of mathematical symbols used in any formulae. Making the effort to fill these tables means the reader has a one-stop place to refer to instead of searching the internet and references to try and find out what you meant by certain abbreviations or symbols.

The list of symbols is split into the Roman and Greek alphabets. Whereas the abbreviations and symbols ought to be listed in alphabetical order (and this is *not* done automatically for you) the list of physical constants should be grouped into similar themes.

The next page contains a one line dedication. Who will you dedicate your thesis to?

Finally, there is the block where the chapters are included. Uncomment the lines (delete the `%` character) as you write the chapters. Each chapter should be written in its own file and put into the *Chapters* folder and named `Chapter1`, `Chapter2`,

etc...Similarly for the appendices, uncomment the lines as you need them. Each appendix should go into its own file and placed in the *Appendices* folder.

After the preamble, chapters and appendices finally comes the bibliography. The bibliography style (called *authoryear*) is used for the bibliography and is a fully featured style that will even include links to where the referenced paper can be found online. Do not underestimate how grateful your reader will be to find that a reference to a paper is just a click away. Of course, this relies on you putting the URL information into the BibTeX file in the first place.

1.11 Thesis Features and Conventions

To get the best out of this template, there are a few conventions that you may want to follow.

One of the most important (and most difficult) things to keep track of in such a long document as a thesis is consistency. Using certain conventions and ways of doing things (such as using a Todo list) makes the job easier. Of course, all of these are optional and you can adopt your own method.

1.11.1 Printing Format

This thesis template is designed for double sided printing (i.e. content on the front and back of pages) as most theses are printed and bound this way. Switching to one sided printing is as simple as uncommenting the *oneside* option of the `documentclass` command at the top of the `main.tex` file. You may then wish to adjust the margins to suit specifications from your institution.

The headers for the pages contain the page number on the outer side (so it is easy to flick through to the page you want) and the chapter name on the inner side.

The text is set to 11 point by default with single line spacing, again, you can tune the text size and spacing should you want or need to using the options at the very start of `main.tex`. The spacing can be changed similarly by replacing the *singlespacing* with *onehalfspacing* or *doublespacing*.

1.11.2 Using US Letter Paper

The paper size used in the template is A4, which is the standard size in Europe. If you are using this thesis template elsewhere and particularly in the United States, then you may have to change the A4 paper size to the US Letter size. This can be done in the margins settings section in `main.tex`.

Due to the differences in the paper size, the resulting margins may be different to what you like or require (as it is common for institutions to dictate certain margin sizes). If this is the case, then the margin sizes can be tweaked by modifying the values in the same block as where you set the paper size. Now your document should be set up for US Letter paper size with suitable margins.

1.11.3 References

The `biblatex` package is used to format the bibliography and inserts references such as this one [**Reference1**]. The options used in the `main.tex` file mean that the in-text citations of references are formatted with the author(s) listed with the date of the publication. Multiple references are separated by semicolons (e.g. [**Reference2**, **Reference1**]) and references with more than three authors only show the first author

with *et al.* indicating there are more authors (e.g. [**Reference3**]). This is done automatically for you. To see how you use references, have a look at the `Chapter1.tex` source file. Many reference managers allow you to simply drag the reference into the document as you type.

Scientific references should come *before* the punctuation mark if there is one (such as a comma or period). The same goes for footnotes[1]. You can change this but the most important thing is to keep the convention consistent throughout the thesis. Footnotes themselves should be full, descriptive sentences (beginning with a capital letter and ending with a full stop). The APA6 states: "Footnote numbers should be superscripted, [...], following any punctuation mark except a dash." The Chicago manual of style states: "A note number should be placed at the end of a sentence or clause. The number follows any punctuation mark except the dash, which it precedes. It follows a closing parenthesis."

The bibliography is typeset with references listed in alphabetical order by the first author's last name. This is similar to the APA referencing style. To see how LaTeX typesets the bibliography, have a look at the very end of this document (or just click on the reference number links in in-text citations).

A Note on bibtex

The bibtex backend used in the template by default does not correctly handle unicode character encoding (i.e. "international" characters). You may see a warning about this in the compilation log and, if your references contain unicode characters, they may not show up correctly or at all. The solution to this is to use the biber backend instead of the outdated bibtex backend. This is done by finding this in `main.tex`: *`backend=bibtex`* and changing it to *`backend=biber`*. You will then need to delete all auxiliary BibTeX files and navigate to the template directory in your terminal (command prompt). Once there, simply type `biber main` and biber will compile your bibliography. You can then compile `main.tex` as normal and your bibliography will be updated. An alternative is to set up your LaTeX editor to compile with biber instead of bibtex, see here for how to do this for various editors.

1.11.4 Tables

Tables are an important way of displaying your results, below is an example table which was generated with this code:

```
\begin{table}
\caption{The effects of treatments X and Y on the four groups studied.}
\label{tab:treatments}
\centering
\begin{tabular}{l l l}
\toprule
\tabhead{Groups} & \tabhead{Treatment X} & \tabhead{Treatment Y} \\
\midrule
1 & 0.2 & 0.8\\
2 & 0.17 & 0.7\\
3 & 0.24 & 0.75\\
4 & 0.68 & 0.3\\
\bottomrule\\
\end{tabular}
\end{table}
```

TABLE 1.1: The effects of treatments X and Y on the four groups studied.

Groups	Treatment X	Treatment Y
1	0.2	0.8
2	0.17	0.7
3	0.24	0.75
4	0.68	0.3

You can reference tables with `\ref{<label>}` where the label is defined within the table environment. See `Chapter1.tex` for an example of the label and citation (e.g. Table **??**).

1.11.5 Figures

There will hopefully be many figures in your thesis (that should be placed in the *Figures* folder). The way to insert figures into your thesis is to use a code template like this:

```
\begin{figure}
\centering
\includegraphics{Figures/Electron}
\decoRule
\caption[An Electron]{An electron (artist's impression).}
\label{fig:Electron}
\end{figure}
```

Also look in the source file. Putting this code into the source file produces the picture of the electron that you can see in the figure below.

Sometimes figures don't always appear where you write them in the source. The placement depends on how much space there is on the page for the figure. Sometimes there is not enough room to fit a figure directly where it should go (in relation to the text) and so LaTeX puts it at the top of the next page. Positioning figures is the job of LaTeX and so you should only worry about making them look good!

Figures usually should have captions just in case you need to refer to them (such as in Figure **??**). The `\caption` command contains two parts, the first part, inside the square brackets is the title that will appear in the *List of Figures*, and so should be short. The second part in the curly brackets should contain the longer and more descriptive caption text.

The `\decoRule` command is optional and simply puts an aesthetic horizontal line below the image. If you do this for one image, do it for all of them.

LaTeX is capable of using images in pdf, jpg and png format.

1.11.6 Typesetting mathematics

If your thesis is going to contain heavy mathematical content, be sure that LaTeX will make it look beautiful, even though it won't be able to solve the equations for you.

The "Not So Short Introduction to LaTeX" (available on CTAN) should tell you everything you need to know for most cases of typesetting mathematics. If you need

[1] Such as this footnote, here down at the bottom of the page.

FIGURE 1.1: An electron (artist's impression).

more information, a much more thorough mathematical guide is available from the AMS called, "A Short Math Guide to LaTeX" and can be downloaded from: ftp://ftp.ams.org/pub/tex/doc/amsmath/short-math-guide.pdf

There are many different LaTeX symbols to remember, luckily you can find the most common symbols in The Comprehensive LaTeX Symbol List.

You can write an equation, which is automatically given an equation number by LaTeX like this:

```
\begin{equation}
E = mc^{2}
\label{eqn:Einstein}
\end{equation}
```

This will produce Einstein's famous energy-matter equivalence equation:

$$E = mc^2 \tag{1.1}$$

All equations you write (which are not in the middle of paragraph text) are automatically given equation numbers by LaTeX. If you don't want a particular equation numbered, use the unnumbered form:

```
\[ a^{2}=4 \]
```

1.12 Sectioning and Subsectioning

You should break your thesis up into nice, bite-sized sections and subsections. LaTeX automatically builds a table of Contents by looking at all the `\chapter{}`, `\section{}` and `\subsection{}` commands you write in the source.

The Table of Contents should only list the sections to three (3) levels. A `chapter{}` is level zero (0). A `\section{}` is level one (1) and so a `\subsection{}` is level two (2). In your thesis it is likely that you will even use a `subsubsection{}`, which is level three (3). The depth to which the Table of Contents is formatted is set within `MastersDoctoralThesis.cls`. If you need this changed, you can do it in `main.tex`.

1.13 In Closing

You have reached the end of this mini-guide. You can now rename or overwrite this pdf file and begin writing your own `Chapter1.tex` and the rest of your thesis. The easy work of setting up the structure and framework has been taken care of for you. It's now your job to fill it out!

Good luck and have lots of fun!

Guide written by —
Sunil Patel: www.sunilpatel.co.uk
Vel: LaTeXTemplates.com

Chapter 2

Background

This section will present the main concepts that are relevant to the thesis project and the reason why it's useful. It will first explain the main theory behind microservices and how it seeks to solve the main problems in legacy software architectures. Then it will detail the new issues that arise from microservices. Finally, it will paint a picture of today's landscape of distributed software and tie it all together to explain the value of the research in this project.

2.1 Microservices

2.1.1 The monolith

In general when talking about both microservices and distributed computing in general, the concept of a **Monolith** often gets brought up. The monolith is this concept of a large, self-contained application where all the code and functionality is tightly bundled together, and is very hard to separate into individual parts. The definition has changed over time: According to the ITS back in 2001 **ITS**, a monolith was "An application in which the user interface, business rules, and data access code is combined into a single executable program and deployed on one platform." However, most modern interpretations online refer back to a book from 2003 called *The art of Unix programming*. Usually referred to as "monster monoliths", it marks the beginning of the trend of using the term monolith as a bogeyman of unmaintainable, poorly planned code. This trend has been continued in modern days with software "gurus" and the like when needing something to compare to our lord and savior microservices**Gouigoux2017**. It is therefore important to remember that: 1: The monolith is not a defined software architecture design paradigm. Rather, it is the default type of application that arises when not taking great effort and intentionality to split the code up in many distinct parts. 2: The monolith is not some damnable evil to be conquered: On software projects of a less-than-huge scale, very often is quite optimal. It doesn't have to deal with inter-component communication. Keeping everything contained in one code base makes it easy to run and test on local machines, keeps all the code in one place for easy access, and so on. 3: The monolith mostly just exists as a concept when talking about microservices. Its purpose is mainly to demonstrate the benefits of microservices in large software.

With that out of the way, let's discuss the problems with monolithic software that microservices seek to amend.

2.1.2 The problems with monolithic software

Methodology

2.2 Imputation

2.2.1 Background

Most relevant algorithms for time series classification do not accommodate missing values. Both the .csv data format utilized for storing the data and some internal data representations employed by sktime and sklearn for processing do not account for missing values. These missing values are represented as "NaN", which stands for "Not a Number". In this document, we will refer to these missing values as NaNs.

To address this issue, several methods can be employed:

- After collecting all the data, remove any columns containing NaNs. This approach results in data loss but ensures the accuracy of all remaining values, as it avoids estimation.
- Limit algorithm usage to only those capable of handling NaNs. A considerable number of algorithms in sklearn and sktime can work in this manner, but it still imposes a constraint **Scikit-learn-imputation**.
- Implement imputation of missing values, which entails using an algorithm to make an informed guess about a plausible value based on existing values in similar positions.

Initially, the first method was effective for the project. This was due to the simulation being poorly configured and not subjected to significant stress. Additionally, there were insufficient time points collected and an inadequate number of instances generated. This meant that there were few opportunities for NaNs to appear in the collected data, so few columns had to be discarded. When the stress testing improved to be more stressful and more data points were gathered, significantly more NaNs appeared and quite a few columns would have to be discarded, inflicting significant data loss. The second method was briefly considered, but since the project's primary goal was to compare multiple classification methods, this idea was quickly discarded.

Ultimately, imputation emerged as the most suitable solution. Imputation of univariate datasets is typically straightforward: Missing values are assigned the mean, median, or most frequent value for their respective column. Multivariate imputation is more challenging. The primary issue is that each column may exhibit significant variation between instances. Simply using a statistic about the entire column would dilute the data, thereby worsening the signal-to-noise ratio.

2.3 Series length

The data collection program that collects the Prometheus data and saves it as .csv files strives to obtain the same number of time points for each instance. This is achieved by using consistent timeframes and collection intervals. However, this is not always possible.

Factors such as the test machine's poor performance under heavy load or data loss during the cleaning process may cause slight variations in the number of data points or time points between some instances. This issue presents a challenge for

statistical classifiers, as most of them are designed to work with datasets of equal length.

There are two main ways to deal with this problem:

- Use only algorithms that can handle series of unequal length. In sklearn/sktime this amounts to exactly two classifiers, a KNN classifier and an SVC (Support vector classifier). They will be used for comparison, but having more options to compare would be better.
- Perform

2.4 Feature selection

Appendix A

Frequently Asked Questions

A.1 How do I change the colors of links?

The color of links can be changed to your liking using:

`\hypersetup{urlcolor=red}`, or
`\hypersetup{citecolor=green}`, or
`\hypersetup{allcolor=blue}`.

If you want to completely hide the links, you can use:

`\hypersetup{allcolors=.}`, or even better:
`\hypersetup{hidelinks}`.

If you want to have obvious links in the PDF but not the printed text, use:

`\hypersetup{colorlinks=false}`.