Georgiana Corduneanu

Resource Usage Forecast

1 2 9 0 UNIVERSIDADE D COIMBRA

UNIVERSIDADE D
COIMBRA

Georgiana Corduneanu

Resource Usage Forecast

Dissertation in the context of the Master's in Informatics Engineering, Specialization in Data Science, advised by Professor João Correia and Professor Penousal Machado and presented to the
Department of Informatics Engineering of the Faculty of Sciences and Technology of the University of Coimbra.

September 2022

FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE D
COIMBRA

DEPARTMENT OF INFORMATICS ENGINEERING

Georgiana Corduneanu

Resource Usage Forecast

Dissertation in the context of the Master in Informatics Engineering, specialization in Engineering and Data Science advised by Professor João Correia and Professor Penousal Machado and presented to the Department of Informatics Engineering of the Faculty of Sciences and Technology of the University of Coimbra.

September 2022

FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE D
COIMBRA

DEPARTAMENTO DE ENGENHARIA INFORMÁTICA

Georgiana Corduneanu

Previsão de Recursos

Dissertação no âmbito do Mestrado em Engenharia Informática, especialização em Engenharia e Ciência dos Dados, orientada pelo Professor Doutor João Correia e Professor Doutor Penousal Machado e apresentada ao Departamento de Engenharia Informática da Faculdade de Ciências e Tecnologia da Universidade de Coimbra.

Setembro 2022

Acknowledgements

First of all, I would like to thank my supervisors, João Correia and Penousal Machado, for their support throughout this dissertation. They handed me the liberty and responsibility to investigate what I truly wanted, offering me a unique opportunity for academic growth.

The Board of European Students of Technology was one of the most incredible experiences I had the pleasure of being a part of during the first four years. It contributed to most of the soft skills I acquired during my time at this University and allowed me to meet extraordinary and broad-minded people all over Europe. I'm thankful for all the incredible challenges it put me through and the personal growth it provided me.

To all my friends with whom I shared the last six years, thank you for all the wonderful memories. From the late-hour projects and stressful exam seasons to the outlandish travels and unique academic traditions, I am grateful that I could share them all with you. I hope we'll continue to be in each other's lives and that, wherever life takes us, we will always celebrate each other's achievements.

Finally, to my family, especially to Mariana, who helped me be the person I am today and remember that "Everything in Life has a Motive". I am very grateful for you and the family that took me in as one of their own.

This work is funded by the project POWER (grant number POCI-01-0247-FEDER070365), co-financed by the European Regional Development Fund (FEDER), through Portugal 2020 (PT2020), and by the Competitiveness and Internationalization Operational Programme (COMPETE 2020).

Abstract

Telecommunication is one of the most fundamental parts of our daily basis, so it proliferates. Therefore the telecommunication hardware and software infrastructure must fulfill this demand and adapt to any situation. To comply with that, one of the methods to optimize telecom services is by forecasting resource usage to meet the client's demand and reduce infrastructural costs. AlticeLabs, over the past two decades, has developed and evolved a solution for real-time control and charging of telecommunications services. At the start, it was called the NGIN - Next Generation Intelligent Network. This work has explored existing research that meets similar goals to the resource forecast. Two forecasting methods were studied, traditional time series forecasting and machine learning adaptation to forecast one selected resource based on the rest of the features provided by AlticeLabs. For each forecasting model, different training sizes and hyperparameters were systematically experimented with to achieve a lower error with lower training time. The best model overall was Seasonal Autoregressive Integrated Moving Average (SARIMA) which has reached an overall Mean Absolute Percentage Error (MAPE) for all machines at the CPN site of 10.69%. This dissertation proved that forecasting could be done using traditional time series algorithms assuming that future value is related to the historical pattern or machine learning adaptations assuming the predicting values correlate with variables. When forecasting CPU Load using Long Short-Term Memory (LSTM) or SARIMA resulted in a lower error compared to Triple Exponential Smoothing or Holt-Winters (TES). As expected for LSTM, using multiple features instead of just the target data results in a mean of 0.21% decrease in error. When comparing methods to forecast future values related to history patterns SARIMA perform better compared to TES. Whereas in the case of SARIMA, the past observations are weighted equally, exponential functions like TES are used to assign exponentially decreasing weights over time. When analyzing the best TES, SARIMA, and LSTM model on the mean for all machines at CPN site SARIMA results in an improvement of 1.3% when compared to TES and an improvement of 0.64% when compared to LSTM. The goals of this dissertation were achieved by developing a forecasting framework that starts by pre-processing data, applying the best-suited forecasting models, and uploading them into a dashboard using a visualization tool.

Keywords

Machine Learning, Time Series, Resources Forecast, Data Mining, Neural Network, Visualization

Resumo

As telecomunicações são uma das partes mais fundamentais da nossa base diária, e por isso, crescem a um ritmo muito elevado. Por conseguinte, a infraestrutura de hardware e 'software' de telecomunicações precisa de satisfazer esta procura e adaptar-se a qualquer situação. Para o cumprir, um dos métodos para otimizar os serviços de telecomunicações é a previsão da utilização de recursos para satisfazer a procura do cliente e reduzir as infraestruturas custos. A AlticeLabs desenvolveu e desenvolveu nas últimas duas décadas uma solução para o controlo e cobrança em tempo real dos serviços de telecomunicações, que no início se chamava NGIN - Next Generation Intelligent Network. Este trabalho explorou a investigação existente que cumpre objetivos semelhantes em contemplação com a previsão de recursos. Foram estudados dois métodos de previsão, a previsão tradicional de séries cronológicas e a adaptação da aprendizagem de máquinas à previsão de um recurso selecionado com base no resto das características fornecidas pelo AlticeLabs. Para cada modelo, diferentes tamanhos de formação e hiperparâmetros foram sistematicamente experimentados para se conseguir um erro menor com menor tempo de formação. O melhor modelo em geral foi a Média Móvel Integrada Sazonal Autoregressiva (SARIMA), que atingiu uma Média Absoluta Global Erro percentual (MAPE) para todas as máquinas no sítio do CPN de 10,69%. Esta dissertação provou que a previsão pode ser feita usando algoritmos tradicionais de séries temporais assumindo que o valor futuro está relacionado com o padrão histórico ou adaptações de aprendizagem da máquina, assumindo que os valores de previsão estão correlacionados com variáveis. Ao prever a carga da CPU utilizando Memória de Curto Prazo Longo (LSTM) ou SARIMA resultou num erro menor em comparação com o Triple Exponential Smoothing ou Holt Winters (TES). Como esperado para a LSTM usando múltiplas características, em vez de apenas os dados alvo, resulta numa média de 0,21% de diminuição do erro. Ao comparar métodos com previsão de valores futuros relacionados com padrões históricos, SARIMA tem melhor desempenho em comparação com a TES. Enquanto no caso da SARIMA as observações passadas são igualmente ponderadas, funções exponenciais como a TES são utilizadas para atribuir pesos exponencialmente decrescentes temporalmente. Ao analisar o melhor modelo TES, SARIMA, e LSTM sobre a média de todas as máquinas no sítio CPN, SARIMA resulta numa melhoria de 1,3% quando comparado à TES e uma melhoria de 0,64% em comparação com a LSTM. Os objectivos desta dissertação foram alcançados através do desenvolvimento de um quadro de previsão que começa pelo pré-processamento de dados, aplicando os modelos de previsão mais adequados, e carregando-os para um painel de instrumentos utilizando uma ferramenta de visualização.

Palavras-Chave

Aprendizagem de Máquina, Séries Temporais, Previsão de Recursos, Mineração de dados, Rede Neural, Visualização

Contents

Acronyms

ACS Autocorrelation. 71, 72, 73, 74

ADF Augmented Dickey–Fuller. 72, 73

AI Artificial Intelligence. 10

ANN Artificial Neural Network. 18, 20

AR Autoregressive. 12, 19, 23

ARIMA Autoregressive Integrated Moving Average. 11, 12, 19, 20, 23

ARMA Autoregressive Moving Average. 19, 20, 23

BJ Box Jenkins. 11, 12, 71, 98

BP Backpropagation. 19, 21

CC Correlation Criteria. 14

CMA-ES Covariance Matrix Adaptation Evolutionary Strategy. 21

CNN Convolutional Neural Network. 18, 20

CPN Central Processing Node. 6, 7, 8, 31, 32, 41, 44

DA-RNN Dual-stage Attention-based Recurrent Neural Network. 20

DE Differential Evolution. 19, 21

DES Double Exponential Smoothing. 19, 23

DNN Deep Neural Network. 20

DSGW Diameter Gy. 6, 7, 31, 32, 39, 40, 41, 44

DyFor GP Dynamic Forecasting Genetic Program. 23

ER Error Rate. 17

ES Exponential Smoothing. 12

FT Fourier Transformation. 19, 21

GA Genetic Algorithm. 14, 18, 20

GBRT Gradient Boosting Regression Tree. 18

GBT Gradient Boosting Tree. 19, 22

GP Genetic Programming. 10, 19, 23

KPI Key Performance Indicators. 27

KR Krigin. 22

LOCF Last Observation Carried Forward. 35

LR Linear Regression. 19, 21, 22

LSTM Long Short-Term Memory. ix, 10, 18, 19, 20, 21, 22, 23, 25, 57, 61, 64, 66, 71, 74, 77, 84, 90, 93, 94, 98, 99

MA Moving Average. 12, 19, 21

MAE Mean Absolute Error. 16, 18, 19, 20, 21, 22, 23

MAPE Mean Absolute Percentage Error. ix, xxiv, xxv, 16, 18, 19, 20, 23, 25, 52, 58, 64, 65, 66, 68, 69, 70, 74, 75, 76, 77, 78, 79, 83, 84, 85, 86, 88, 90, 93, 94, 97, 98, 99

MAR Missing At Random. 15

MCAR Missing Completely At Random. 15

MER Mean Error Rate. 17, 19, 21

MI Mutual Information. 14

ML Machine Learning. 10, 14, 18, 93, 97

MM Mean error of the Mean values. 17, 19, 21

MNAR Missing Not At Random. 15

MSE Mean Square Error. 16, 19, 20, 21, 25, 58, 59, 64, 66, 71, 75, 97, 98

NGIN Next Generation Intelligent Network. 3, 29, 30, 99

NGIN-CCP Next Generation Intelligent Network - Convergent Charging and Policy. 5, 6, 8

NN Neural Network. 20, 21

NWS Network Weather Service. 21

PACS Partial-autocorrelation. 71, 72, 73, 74

PCA Principal Component Analysis. 20

PSO Particle Swarm Optimization. 14, 21

RF Random Forest. 14, 18, 19, 20, 22, 61, 93, 98

RMSD Root-Mean-Square Deviation. 17, 19

RMSE Root Mean Square Error. 16, 18, 19, 20, 22, 23

RNN Recurrent Neural Network. 2, 10, 19, 20, 21, 22, 64

RR Ridge Regression. 14, 18, 20

RWF Random Walk Forecasting. 21

SARIMA Seasonal Autoregressive Integrated Moving Average. ix, 11, 12, 19, 23, 25, 61, 71, 76, 77, 84, 86, 90, 93, 94, 97, 98, 99

SD-MER Standard Deviation of MER. 17, 19, 21

SGD Stochastic Gradient Descent. 20

SMP Service Management Platforms. 6, 7, 8, 31, 32

SVM Support Vector Machines. 19, 20, 22

SVR Support Vector Regression. 14, 18, 20

TB Tendency Based. 19

TES Triple Exponential Smoothing or Holt-Winters. ix, 19, 23, 25, 61, 71, 82, 84, 85, 86, 90, 93, 94, 97, 98, 99

TS Time Series. 10, 12, 21, 28, 71

List of Figures

List of Tables

Chapter 1

Introduction

Telecommunications are the means for information to travel at a distance. This information may come from voice, message, data, text, images, or video. Some technologies, such as wire, optical fiber, radio, or electromagnetic systems, can transmit this information. Possible at a global scale, it can be used with a phone, a computer, wired or wireless, with or without internet, making it a very competitive area, so there is a need for improving services aiming at the held of the customer. There is a clear focus on resources when talking about the demand and investment in infrastructure or software to handle it depending on the type of request since internet access is growing at a high rate compared to voice calls or SMS, which has shown a decrease in demand.

Nevertheless, the use of telecommunication is growing. Unfortunately, there is no consistency when it comes to its usage. Sometimes, the user is more frequent, like Christmas or New Year, or even at a smaller granularity. There are times of the day when the use decay, when sleeping, for example. Predicting the usage of the service is crucial to responding to the client's request. When there is no response, or it is too, it can create a disappointment to clients and therefore influence others negatively. A software that can predict these peaks not only when there is a high demand but also when there is low demand can optimize the company's service by decreasing its costs, for example.

AlticeLabs has developed and evolved a solution for real-time control and charging of telecommunications services over the past two decades, which started to be called NGIN - Next Generation Intelligent Network. This solution can assist most of the needs of the company. However, this tool needs to be developed regularly to keep up with updates in the real world.

The process of extracting valuable and valid information from a large amount of data is termed Data mining [1]. Data mining can extract useful information or detect patterns to analyze and make future predictions. These predictions can help companies or organizations make better business growth decisions. For example, resource usage forecast meaning is predicting the future of resource requirements to monitor and analyze these systems. To do so, we need to discover interesting patterns in the database that are useful for decision support or even alarm triggering. These patterns help plan strategies to prevent problems or decrease the

cost of unused resources.

Data is available in different formats and structures. In the case of this dissertation, temporal data is a type of data that varies over the change in time, it is represented using time stamps, and the process of finding patterns is called temporal data mining [1].

Time Series Data is a sequence of data points that occur in a temporal order over some time. Its granularity depends on how the company or organization collects data. It can be represented in several ways: per year, month, day, or hour.

The workload in the form of CPU utilization often fluctuates, which leads to unnecessary cost and environmental impact for companies [2]. It is a challenging task studied for years to optimize resource management. Although the activity of making this task automated is recent, traditionally, it was a task for the business analyst, generally using statistical techniques. Neglecting the importance of resource management can lead to business or client harm. On the other hand, excessive monitoring can lead to the system's overhead and, thus, increase the cost associated with analyzing the network infrastructure.

In a distributed computing environment, resources, such as CPU, memory, disks, Etc., are time-shared by many tasks. Generally, a scheduling strategy is applied to guide the execution of these tasks to achieve high performance. Resource monitoring and usage prediction are required [3].

Previous researchers have proven that using a Recurrent Neural Network (RNN) can be forecasted CPU utilization for the next 15 minutes [4] or 20 minutes [5]. However, depending on the company's need, a maximum of 20 minutes can be too small, and a longer lag into the future may be necessary.

With this in mind and having access to historical data, some machine learning or time series techniques can be applied to forecast essential resources and ease the telecom system. For example, when we are in the presence of a low peak or even provide enough resources, the system will be prepared for higher peaks.

In the case of AlticeLabs, as it was said, it produces two types of data, operational and business, and our goal is to take advantage of both to add value to the forecasting. To that end, there are some questions to start with:

1. Do CPU Load evolve in the same way with time for all the machines in the telecom ecosystem?

2. How do waiting processes evolve over time?

3. Which are the business features with the strongest influence over the operational data?

4. What are the days with the highest and lowest requests? Which type of request influenced most?

5. What is the best forecasting model?

6. Can an automated tracking visualization be created for business features and forecasting data?

Considering all these questions and AlticeLab's needs, this dissertation aims to create a prototype to answer all these questions and forecast the essential feature to help optimize telecom resources.

1.1 Motivation

Computational power is often over-provisioned to sustain workload during peak hours which generates the idle capacity outside peak hours [2]. By optimizing computational resources and avoiding over-utilization or under-utilization, it would be possible to reduce both costs. One way to do this is to provide a tool that reduces the uncertainty of future resource utilization.

AlticeLabs has created a product called Next Generation Intelligent Network (NGIN), by extracting reports from this tool we can improve this tool by adding a forecasting piece. NGIN tool is capable of dealing with the requests and keeping a record of them, but it can be improved by forecasting the usage of these resources and so benefiting from the problem-solving or cost-reducing that it can provide.

By adding a new feature to NGIN, forecasting resources, the next NGIN will be able to suggest when high or low peaks may happen, or correlate business data and adapt resources or eventually marketing strategies. Further on will be shown the created prototype of this forecasting upgrade and how can it be useful within the telecom environment.

There is a need to keep up with the demand and be prepared for any situations and daily improving tools that we work with to evolve at the same time as the local reality needs.

1.2 Outline

This document has the following structure: a state of the art (Chapter 2) where the reader will understand the context, data, the problem, and related work focused on feature importance and selection and forecasting; and an approach (Chapter 3) where the reader will understand the requirements, framework, motivation of the methodologies and the initial the pre-processing; the forecasting (Chapter 4) where all the methodologies and approaches in order to forecast will be explained; forecasting results afterwards (Chapter 5) where it is shown how data behaves when applying the forecasting methodologies, the subsequently forecasting and the final dashboard; a planning chapter (Chapter 6) where it will be discussed how the implementation of the dissertation occurred compared to the initial proposal, a conclusion and future work.

Chapter 2

Context and State of the Art

As it was already said and studied in the scientific community, network traffic forecasting is a benefit for a company with the goal of reducing costs to give clients a better user experience.

There are multiple approaches to forecast resources. For example, it can be done using historical data, real-time, or for a long step ahead window. In the following subsections, some approaches are exhibited to deal with past and future values and the best algorithms for each case.

First, dataset context and content will be described. Second is the problem definition in the AlticeLabs environment. Third is the background knowledge necessary to examine the work and related work with new eyes, and finally, a summary section.

2.1 Dataset Context and Definition

Next Generation Intelligent Network - Convergent Charging and Policy (NGIN-CCP) is a business intelligence solution that gives a complete and detailed overview of the NGINPCC solution in the form of two types of reports: operational data which is considered to be the inventory and business data and business data which is considered to be the usage-related activity.

This report gives telecom operators a complete and detailed vision of the solution status and usage activity, as described in the following subsections.

2.1.1 Business Data

The usage-related reports are based on periodically receiving activity registers from all the NGIN-CCP modules, guaranteeing near real-time reporting. Information is organized accordingly with the primary process that generated it. The processes cover the complete NGIN-CCP activity, typically from provision to service usage or topup 3.25.

Business Data Size			
Data/D	Size/D	Data/M	Size/M
24*5=120	≈ 0.5KB	3 600	≈ 14.5KB

Table 2.1: Business Data Size - Number of data cells (number of hours*number of features) and its size per day(D) and month(M)

- SMS shows the total number of SMS events.
- Call shows the total number of voice call events.
- Data shows the total number of finished data session events.
- Provisioning shows the total number of provisioning operations like create, remove, modifications
- Topup shows the total amount of credit operations, like debit or recharges.

In order to have the business dataset, all its features are merged on the feature data for SMS, voice call, data session, topup, and provisioning. This process will result in only one dataset characterized by its five features and data per day, as seen in table 2.1.

2.1.2 Operational Data

The base for Inventory, for a status reporting of the solution entities, is a daily photo taken of the entities that compose NGIN-CCP.

Diameter Gy (DSGW) starts by receiving the charging events per voice, SMS, and data session, which are forwarded to Central Processing Node (CPN) to proceed with executing these. This process ends by receiving the reports of the internal records from Service Management Platforms (SMP) to keep historical evidence.

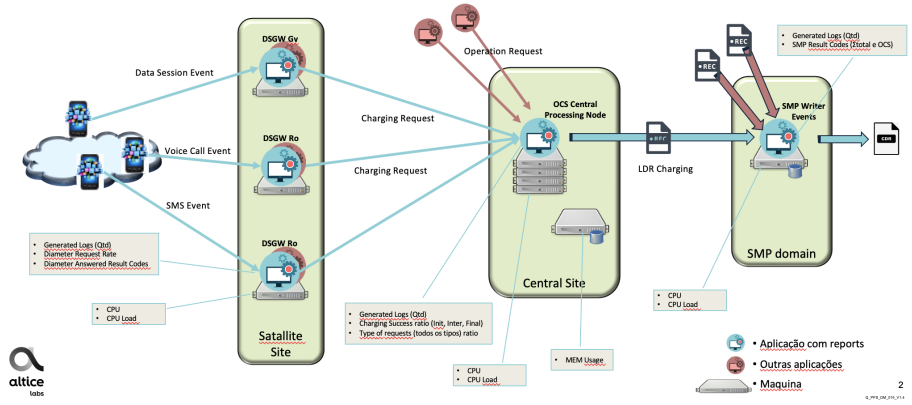


Figure 2.1: Real-Time Charging Process

A telecommunication charging process creates a dataset that presents three charges: data session events, voice call events, and SMS events.

- DSGW, it is located at the Satellite Site, and here the logic is executed to perform protocol adaptation, signaling control, and requests management. DSGW receives requests to perform charging of voice, SMS, data, or specific services
- OCS CPN, where the policy control, rating, and charging functionalities are executed at this site
- SMP Writer Events, receives all events and registers at the internal database

These event requests mentioned before SMS, voice call, and data session are created by a subscription linked to a pack previously bought by a company or for personal use. A company buying a pack requires several subscriptions linked to it. When it is for personal use, then it corresponds to one subscription. These two types of subscriptions, either for a company or personal use, are formerly allied to a DSGW machine, which is processed and forwarded to a CPN machine to be executed.

For the operational data, the process is not as easy considering its three different sites, satellite, central, and SMP domains. They have three different types of reports in common:

- CPU, lower, higher and average percentage of CPU usage each hour per metric (idle, system, user, IO wait, nice, steal, interrupt)
- CPU Load, or average load, lower, higher, average number of processes waiting to be executed summed to the ones already being executed in the last 5 and 15 minutes
- Generated Logs, lower, higher, average, and the absolute number of different types of logs generated by the application per hour (error, warn, info, debug, trace).

Starting by looking in more detail at DSGW receives charging events per voice, SMS, and data session, which is most of the system use. This application has instances for each, but they run deferentially on four machines. At this level, there are two types of reports apart from the three mentioned above:

- Result Codes, lower, higher, average, and the absolute number of requests each hour (success, errors, undefined)
- Request Rate, lower, higher, average, and absolute rate request

After the pre-process at DSGW the requests are forwarded to OCS CPN where it receives and executes events. This application contains 16 machines where events run deferentially. At this level, there is one report apart from the three mentioned above:

Operational Data Size				
Site	Data/D	Size/D	Data/M	Size/M
DSGW	108*24*4=10 368	≈ 78KB	311 040	≈ 2.3MB
CPN	104*24*16=39 936	≈ 286KB	1 198 080	≈ 14.5MB
SMP	64*24*6=9 216	≈ 69KB	276 480	≈ 2MB
TOTAL	64*24*6=59 520	≈ 433KB	1 785 600	≈ 18.8MB

Table 2.2: Operational Data Size - Number of data cells (number of features * hours * number of machines) and its size per day (D) and month (M)

- Type of Requests and Charging Success Ratio, lower, higher, average, and absolute number each hour for the charging events and result of the request (success, errors, undefined)

At the end of the pipeline, the SMP Writer Events receives reports of the internal records from the CPN. The application has six machines running deferentially. At this level, there is only one report apart from the three mentioned above:

- SMP Result Codes (OCS), lower, higher, average, and the absolute number of results each hour per each metric of the requested charging result (success, errors, undefined).

This process will result in three datasets, one per site, which are characterized by their features specified before as as seen in table 2.2. As previously stated, CPN is a site composed of more machines than the others and has many features, making it the site with a greater need for memory space.

2.2 Problem Definition

We have three main goals that will be approached during this dissertation: forecasting resources or a resource that may present some randomness, creating automated visualizations in Grafana, and extracting knowledge from operational and business data.

The problem that is faced is understanding data and its dependencies. We have two perspectives of data resulting from NGIN-CCP, an operational recommended for diagnosing operational problems and the business perspective recommended for macro business information. The goal is that by mixing the knowledge from both sides, we may be able to discover some problems or patterns to reduce resources possibly.

2.3 Concepts and Definitions

The area of machine learning is one of the most prominent subjects in research. Consequently, multiple fields research a way of optimizing their work, service, or processes. Machine learning aims to predict outcomes that approximate the future with as little error as possible with the help of some input using statistics to estimate a function [2]. This function can contain different types of parameters. Some are set before training, and others are updated during training to minimize or maximize the cost function.

This section will start by presenting the main differences between forecasting and prediction used during this dissertation. Right after the different types of machine learning and time series forecasting algorithms and how to use them, from the simpler models to the most complex ones. Subsequently, methods are introduced to select features to input into the models prepared for it and, in the end, how to classify values as outliers to help further characterize a resulting forecast.

Forecasting vs. Prediction		
Comparison	Forecasting	Prediction
Main Difference	Essentially around future outcomes	Not necessarily concerned about the future
Data	Based on historical data find future outcomes	Not based on past comparatively they just give the possibility of the outcome
Output	The output will be a future scenario based on the past	The output are logical analysis in contemplation of the estimation for a certain event
Application	Useful in economical and meteorological fields	Applied in majority of fields as long as there is a need of an explanation of an event in the future

Table 2.3: Comparison between forecasting and prediction

Prediction is concerned with estimating the outcomes for unseen data. A model is fitted to a training set to make predictions for new samples. When predicting, we are not always finding future outcomes, so it is not bounded by any time, past or present.

Forecasting is a branch of prediction applied to time series. Based on historical data, we forecast the future using trend, seasonality, and other time series statis-

tics. Classical time series can be subdivided into five categories:

- Exponential Smoothing Methods
- Regression Methods
- Autoregressive integrated moving average methods
- Threshold methods
- Generalized autoregressive conditionally heteroskedastic methods

The first three listed above can be considered linear methods, and the last two as non-linear methods. See Makridakis et al. [6] for more details about these statistical methods.

Nowadays, some modern heuristic methods for time series forecasting are based on neural networks and methods based on Genetic Programming (GP) [7] that will be shown further on.

Time Series Forecasting Time Series (TS) is a collection of observations made sequentially through time. The objective of TS can be categorized into four domains, description, explanation, prediction, and control. From the description TS can be analyzed and identify trends, missing data, outliers, etc. This knowledge can be used for the optimization of problems or to understand them and perform a prediction.

TS can be decomposed into trend, seasonality, or other cyclic patterns. If they are composed by them then that TS is said to be a non-stationary TS, otherwise it is a stationary TS. This definition is important to define the forecast approaches to be used.

Machine Learning The area of machine learning is one of the most outstanding subjects of research [2], it is a type of Artificial Intelligence (AI) that allows the software to predict outcomes without being explicitly programmed. In addition, Machine Learning (ML) are trained using a training dataset of selected features, and the validation set with new data is used to ensure its generalizability and validate the accuracy of the selected features [8] [9]. This optimization is done by using evaluation metrics that will be talked about in the sub-sections below.

Pre-processing the dataset before training is a significant step when using a ML model [10]. The pre-process may differ depending on the type of ML. For example, in the case of supervised learning, the dataset used requires a label to help the prediction. On the contrary, no labels are given for unsupervised learning, and the algorithms find a structure from input on their own.

Long-Short Term Memory Long Short-Term Memory (LSTM) is an extension of Recurrent Neural Network (RNN) with an improved memory. LSTM can remember inputs over a longer period. This algorithm has three types of gates input, output, and forget. Based on the given information importance these gates decides if it is going to be stored. The importance of the information is based on the weights, which are also learned by the LSTM. The gates responsible for writing,

deleting, and reading are in the form of sigmoids, meaning they range from 0 to 1. The goal of this algorithm is to determine the importance of the information.

Random Forest It can also be called Random Decision Forest, an ensemble learning method that can be used for both classification and regression. This algorithm fits several decision trees classifier on multiple sub-samples of the training data and uses averaging to improve accuracy and decrease overfitting. In the case of regression problems, the algorithms return the mean or average prediction of the individual trees.

Time Series Decomposition Understanding the time series decomposition is important to classify time series as stationary or not and to apply the right model to it. Time series can be composed by three components:

- Trend, behavior over time
- Seasonality, seasonal period behavior
- Residual, everything not captured by trend and seasonality

In the case of applying a stationary model to a non-stationary time series, seasonality and trend are generally removed. Then the model is applied to the remaining data, called residual. Two types of combinations of the three components lead to the final time series. It can be:

- **Additive**, individual components are added together, meaning that each component is independent.

$$y_t = T_t + S_t + R_t \tag{2.1}$$

  When trend is additive, it indicates a linear trend, and additive seasonality indicates the same frequency and amplitude cycles.

- **Multiplicative**, individual components are multiplied together, meaning they depend on each other.

$$y_t = T_t * S_t * R_t \tag{2.2}$$

  When the trend is multiplicative, it indicates a non-linear trend. A multiplicative seasonality indicates cycles' increasing/decreasing frequency and/or amplitude.

- **Pseudo-additive**, it can also be a mixture of additive or multiplicative models.

Box Jenkins Methods Box Jenkins (BJ) is a time series analysis to apply Autoregressive Integrated Moving Average (ARIMA) and Seasonal Autoregressive Integrated Moving Average (SARIMA) to time series forecasting. This method is composed of several steps to analyze data to be forecasted until reaching a set of parameters to be applied to the forecasting model. It is a manual proceeding and highly dependent on the data scientist's experience. Another way of reaching the

parameters is by grid-search. For a more exhaustive study about BJ can be seen in Das [11].

Moving Average (MA) One of the **stationary** TS models is Moving Average (MA) it will be referenced on subsection section 2.4. It consists of predicting a future value by averaging the n previous value. Usually, this method is used as a baseline algorithm, if the one compared to it has worse results then it should be better to switch algorithm [2] as explained on the fourth key point.

Autoregressive Integrated Moving Average (ARIMA) One of the **non-stationary** TS models are ARIMA will be referenced on subsection section 2.4. ARIMA is the combination of Autoregressive (AR) and and MA, "i" corresponds to the integrated differencing (removing trend of data) step. It can be used to understand the data or predict future trends. The algorithm will identify the appropriate amount of differencing to be applied to data and verify if it is stationary. It will then output the results ARIMA is useful in data with high seasonality. The drawback is not being good at detecting outliers [12]. The notation of ARIMA is (p,d,q) where:

- p is the number of autoregressive terms
- d is the number of differences
- q is the number of moving average terms

Seasonal Autoregressive Integrated Moving Average (SARIMA) A temporal random process of SARIMA type is nothing other than a generalization of ARIMA model containing seasonal part [13]. The notation for SARIMA is (p,d,q)(P,D,Q)S, same parameters as ARIMA and seasonal components:

- P is the seasonal autoregressive order
- D is the seasonal difference order
- Q is the seasonal moving average order
- S is the seasonal period

It is essential to highlight that the S parameter influences (P, D, Q) parameters. For example, S=24 for hourly data suggests a seasonal cycle of 1 day. P=1 would use the first seasonally offset observation(o) o=S*1 or o=24. For P=2, it would use the first two, o=[(S*1),(S*2)], or o=[24,48].

Exponential Smoothing (ES) Exponential Smoothing (ES) is a time series forecasting method where the forecasting is a weighted linear sum of past observations. However, the difference with the previous time series forecasting methods is that ES explicitly uses an exponentially decreasing weight for past observations. This means that different weights are given to the recent and oldest observations. There are three most known exponential smoothing time series. A simple method gives weights to the values, another additionally gives weights to the trend, and the most advanced additionally gives weights to the seasonality.

Simple Exponential Smoothing (SES)

Also known as Single Exponential Smoothing, is a time series forecasting method that handles data with no trend and seasonality. Its notation is alpha, called smoothing factor or smoothing coefficient. It controls the rate of decay of past observations exponentially. Often it is set to a value between 0 and 1. Higher values make the model pay attention to the most recent past values, whereas lower values are the opposite when forecasting. Parameters are:

- alpha - smoothing level

Double Exponential Smoothing (DES)

This model can also be called Holt's linear trend model and adds support for trends in univariate time series. Besides the alpha factor, a beta factor is added to control the exponential decrease of the trend called b. Trend can change in two ways, multiplicative when the trend is exponential or additive if the trend is linear. When the forecast is more extended, we may consider dampening the trend over time, meaning that it reduces the size of the trend down to a straight line (no trend). This process can be done in additive or multiplicative ways. The coefficient p is responsible for the decreasing rate of the trend. Parameters are:

- alpha - smoothing level
- beta - smoothing trend
- trend - additive or multiplicative
- dampen - additive or multiplicative
- phi - damping coefficient

Triple Exponential Smoothing (TES)

Also known as Holt-Winters Method, this model is the most advanced exponential smoothing variation. This model, besides the trend support, adds seasonality support. The new parameter is called gamma and controls the decreasing seasonality rate. This seasonality can be modeled as multiplicative or as additive. Moreover, to model the seasonality correctly, the parameter period is added to describe the seasonal period of the data. Parameters are:

- alpha - smoothing level
- beta - smoothing trend
- gamma - smoothing seasonality
- trend - additive or multiplicative
- dampen - additive or multiplicative
- phi - damping coefficient

- seasonality - additive or multiplicative
- period - time steps in seasonal period

Feature Selection Feature selection processes to identify relevant features over not relevant and redundant features from a dataset [10]. This feature selection is used because the forecasting or prediction result ML does not depend only on the models but also on the data. Data can be composed of features in a tabular representation, that means its columns. Therefore, we need to understand data, identify the importance of the features, and later select the ones that result from a model with better accuracy. There are three kinds of feature importance applications, which can be done by a filter, wrapper, or by embedded features selection [9].

- **Filter Method** is the most used because of its relatively low computational cost and good efficiency. It analyses the correlation and redundancy between features and targets. Some methods can be Correlation Criteria (CC) and Mutual Information (MI)
- **Wrapper Method**, become computationally expensive when the feature space dimensions are quite large, and these methods use various search algorithms to enumerate subsets of the original feature space. Some methods can be Genetic Algorithm (GA) and Particle Swarm Optimization (PSO).
- **Embedded Method**, can reduce computational time. Feature selection is part of the training of machine learning. Some can be Random Forest (RF), Ridge Regression (RR), and Support Vector Regression (SVR).

There are multiple approaches to select subsets of features, one can be an exhaustive search over all possible combinations, but depending on the number of features may turn out to be an operation with a high demand of complexity. The next best alternative is the branch, and bound algorithm, which has been shown to perform reasonably well in specific scenarios [14].

The most famous techniques behave in a "markovian" fashion because there is no going back once a certain feature has been excluded or included. There are two forms of doing this:

- **Sequential Search or Forward Selection algorithm** starts with an empty subset and adds one feature at a time, which maximizes prediction performance in an existing subset.
- **Backward Elimination**, here instead of adding a feature, it starts with a subset containing all the features and shrinks the set by removing one feature at a time based on its performance

Missing Data Characterization Missing data are data values that are not stored for a variable in the observation of interest [15]. Researchers deal with missing data all the time. It can reduce the statistical power of a study and produce biased

estimations, which significantly affect the conclusions from the data. We can encounter three types of missing data and, therefore, multiple techniques to handle this missing data.

- **Missing Completely At Random (MCAR)**, the probability of missing data is not related to the set of data. It can happen because of an equipment failure, can be lost in transit, or is technically unsatisfactory.
- **Missing At Random (MAR))**, the probability of missing data is related to the set of data, but it is not related to the missing data itself.
- **Missing Not At Random (MNAR)**, if missing values cannot be characterized by the previous, then it falls in the category of MNAR. This characterization is problematic since a more complex model estimates missing data.

Techniques for Handling with Missing Data The best way to handle missing data is by preventing the problem and collecting data carefully. There are a number of alternatives to handle missing data. Some of them are:

- **Listwise or Case Deletion**, this is the most common approach when handling missing data. For this technique, we omit the missing cases and analyze the rest of the data [15].
- **Pairwise Deletion**, this strategy eliminates when the particular missing data is needed to test a particular assumption. This technique preserves more information than the previous one, but if there are many missing values compared to the length of the dataset, then we may have an unsatisfactory analysis.
- **Mean Substitution**, the mean value of a feature is used in place of the missing values of the same feature. This strategy preserves the continuity of the dataset, and it is a reasonable estimate for a random selection from a normal distribution. However, this approach adds no new information, increases the sample size, and underestimates the error.
- **Last Observation Carried Forward**, replaces every missing value with the last observed from the same feature. This technique is usually used in the time series approach. This technique is advantageous because it is a simple approach to implementing and communicating however this method assumes that the missing values remain unchanged.
- **Regression Imputation**, imputation is the process of replacing the missing data with estimated values. It replaces with a probable estimated value based on other available information. It is a great approach because, unlike listwise or pairwise, deletion does not alter the distribution's standard deviation or shape. [15].

These techniques should only be employed after seeking and understanding multiple reasons for the missing data and applying appropriate techniques to prevent it.

Outlier Detection The goal is to estimate the future values and get the closest possible to the actual value of the peaks. In this case, we can consider these peaks as outliers. For this, there are multiple techniques to characterize outliers in data. In the case of Amidan et al. [16] explains how an outlier may occour:

- Typographical or any human intervention
- Misunderstanding questions (surveys)
- Instrumentation Breakdown or out of calibration
- Merging distinct data with different characterization

It is based on Chebyshev's Theorem, which was designed to determine a lower bound of the percentage of data that exists within k number of standard deviations from the mean The Chebyshev outlier detection uses this inequality to calculate an outlier. This can be obtained by calculating an upper limit and a lower limit, and if a value gets out of this limit bounds, then it is considered an outlier. The following equations can calculate the limits:

$$UpperLimit = \mu + k * \sigma \tag{2.3}$$

$$LowerLimit = \mu + k * \sigma \tag{2.4}$$

Where μ and σ are calculated from data, the higher the k, the less rigorous the calculation of outliers will be.

Regression Metrics Finally, there are several metrics to be used in order to validate a model. Instead of categorizing a discrete input, the output is predicted in a continuous form. The regression method is extensively used [1], and it is no different for this dissertation.

The following metrics contain some of the most common ones and those used in section 2.4.

$$MeanAbsoluteError(MAE) = \sum_{i=1}^{N} |y_i - \hat{y}_i| \tag{2.5}$$

$$MeanSquareError(MSE) = \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 \tag{2.6}$$

$$RootMeanSquareError(RMSE) = \sqrt{\frac{1}{n} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2} \tag{2.7}$$

$$MeanAbsolutePercentageError(MAPE) = \frac{100\%}{n} \sum_{i=1}^{N} \left| \frac{y_i - \hat{y}_i}{y_i} \right| \tag{2.8}$$

$$Root - Mean - SquareDeviation(RMSD) = \sqrt{\frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)}{N}} \tag{2.9}$$

$$Meanerroroftheir Meanvalues(MM) = \frac{1}{N}\sum_{i=1}^{N}\frac{|\overline{\hat{y}} - \overline{y}|}{y} * 100\% \tag{2.10}$$

$$ErrorRate(ER) = \frac{|\hat{y} - y|}{y} * 100\% \tag{2.11}$$

$$MeanErrorRate(MER) = \frac{1}{N}\sum_{i=1}^{N} ER_i \tag{2.12}$$

$$StandardDeviationofMER(SD - MER) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(ER_i - MER)^2} \tag{2.13}$$

2.4 Related Work

This section contains several related works regarding forecasting and feature selection. For every work it will be presented, algorithms used, logic performed, and the final metric used to be compared further. In addition, each one of these topics will have a summary table containing the algorithms used, their author, metrics used, and how many steps ahead the algorithms forecasted for the higher number.

2.4.1 Feature Importance and Selection

Feature selection is applied because it reduces the model's training time, helps simplify its complexity, improves accuracy, and avoids overfitting by eliminating unnecessary variables from the feature set. Some works showed another important aspect of the machine learning process. How many features should we choose and which ones? Chen et al. [9] proposed a test with embedded feature selection. SVR, Gradient Boosting Regression Tree (GBRT) and ML were proposed as forecasting algorithms, as feature selection algorithms were selected RF, RR, SVR as comparison the proposed approach and the evaluation metric used were MAPE, MAE and RMSE. The proposed feature selection used the results of three embedded feature selection algorithms, RF, RR and SVR to aggregate them into a single set by weighted vote, which results in a set of features by aggregation. The results showed that in most of the scenarios, the proposed algorithm dramatically improved over the individual feature selector, and the second best one was RF.

Another work where RF was used as a feature selector was in Lahouar et al. [25], where features were selected to forecast one day-ahead photovoltaic power. Feature importance was executed with RF and was selected by their importance value. As forecasting algorithms were tested RF, Artificial Neural Network (ANN) and persistence (PER) were it assumes that the actual value is the same as the previous. The result was accurate and satisfactory in most cases, with a good ability to handle seasonal effects, and the best forecasting algorithm overall was RF using as evaluation metrics MAPE, MAE and RMSE.

There are other ways to get the importance and selection of the features. Lee et al. [26] has shown this step's importance while experimenting with a forecast with all features and for a selected set of features using a Convolutional Neural Network (CNN). The proposed feature selection was made by fitting a CNN model with each feature and storing its RMSE by testing it with the testing set. The selected features were above a threshold decided by [26]. Furthermore, it was proven that using only the features with higher correlation resulted in better results than all.

Another way of feature selection is using GA, Chen and Zhou [27] also used LSTM to stock prediction model because an ANN can learn any nonlinear relationship and is less disturbed by noise data. Also LSTM is an improved version

Ref.	Technique	Metric	Step Ahead	Lowest Error
[17]	RF, LR, SVM, GBT	MAPE, MAE, MSE	20	3 (MAE)
[18]	LR, SVM, GBT	MAE, MSE	5	2.56 to 14.5 (RMSE)
[19]	TES, HET,DES, ARIMA, ETS	RMSD	1+	2.56 to 14.5 (RMSD)
[5]	PSO, DE, CMA-ES, RWF, MA, LR, BP	MAE, MSE	4	0.23 (MAE)
[12]	ARIMA, TES, LSTM	MAPE,RMSE	1	48 (MAPE)
[3]	FT	MM, MER, SD-MER	1	9.78 to 35.85(MER)
[20]	TB	Mean, RMSE	1	2 to 55 (Mean)
[2]	LSTM, MA,RNN	MSE, MAE	30	0.87 (MAE)
[7]	GP, DyForGP	MSE	1	0.19 to 0.23 (MSE)
[21]	DES, TES	MSE	1	59.16 (MSE)
[22]	ARIMA, TES	MSE,MAPE	1	0.26 to 7.6 (MAPE)
[23]	SARIMA, TES	RMSE	12	0.82 to 1.07 (RMSE)
[24]	AR,ARMA, SARIMA,DES, TES	RMSE	11	2.58 (RMSE)

Table 2.4: Survey of related works to forecasting resources

Ref.	Feature Selection	Metric	Step Ahead	Lowest Error
[9]	RF, RR,SVR	MSE, MAE, MAPE	1	3.1 to 6.72 (MAPE)
[25]	RF, ANN, PER	MSE, MAE, MAPE	1	21 to 28 (RF) (MAPE)
[26]	CNN	RMSE	100	0.18 (MSE)
[28]	DNN	MAPE	1	2.05(MAPE)
[27]	PCA, GA	MSE	5	39 to 53(MSE)

Table 2.5: Survey of related works to feature selection

of RNN because of its memory. It was concluded that models applied to selected features using GA and LSTM as predictor delivered the best results. Forecasting using all the features with RF and LSTM resulted in worse models. Between models with feature selection there were Principal Component Analysis (PCA) combined with Support Vector Machines (SVM) and Dual-stage Attention-based Recurrent Neural Network (DA-RNN) with higher error than using GA and LSTM. When the control parameters of GA are set, such as crossover rate, mutation rate, and the number of factor combinations, a variety of suitable combinations can be derived to improve the research performance.

Chang and Tsai [28] proposed a solution to forecast the number of tourists since it is an indicator of the tourism demand and can serve as the reference for the government policies about tourism and the business strategies of tourism industries. A proposed associative model like SVM, Neural Network (NN), and a deep learning neural network with feature selection can forecast this number. An extra feature selection was made by eliminating redundant and irrelevant features. Data is split into 80% for training and 20% for testing. Features are once again selected based on a GA, and the result is used to train the proposed algorithms. For the NN there were tested different activation functions like sigmoid function and ELUs and optimizers like gradient descent and stochastic gradient descent (SGD). Testing for different epochs showed that as activation function ELUs has the best performance and for optimizer was Stochastic Gradient Descent (SGD). In the end, the best algorithms were the ones with feature selection, and by a small percentage, deep learning has better results.

2.4.2 Forecasting Algorithms

Forecasting algorithms as explained on Table 2.3 use historical data in order to essentially around future outcomes. Here will be presented traditional time series algorithms and machine learning adaptations in order to forecast these future outcomes.

Mason et al. [5] explains that methods like ARIMA or Autoregressive Moving Average (ARMA) rely on patterns in historical data to forecast and, therefore they are not suitable when there are no distinct patterns in data or if there is a signification amount of random variation in data. For this reason Mason et al. [5] proposes a NN approach due to its robustness by not needing assumptions or

requirements about the dataset when using it. Algorithms based on NN used are PSO, Differential Evolution (DE) and Covariance Matrix Adaptation Evolutionary Strategy (CMA-ES). In order to prove this theory Mason et al. [5] compares NN with Random Walk Forecasting (RWF), MA, Linear Regression (LR), Backpropagation (BP). Metrics used to decide best algorithms are MAE and MSE, he proves that non evolutionary algorithms (BP, RWF, MA and LR) even though the best one between neural network-based algorithms is CMA-ES. Mason et al. [5] proved that evolutionary algorithms perform better on unseen data which confirms the generality of trained neural networks. He took it further by experimenting on multi-step ahead prediction trying to know how far in the future with good accuracy can the CPU be predicted or if the model is generalized enough to be used in multiple hosts. As planned, the accuracy decreases more and more when we predict into the future and the evolved networks are capable of accurately predicting CPU utilization on unseen data from both the same or new hosts, which demonstrates a high degree of generalization.

Forecasting future system performance can be used to guide adaptations to their behavior in response to changes in system status [20], so Yang et al. [20] implemented several one-step-ahead and low-overhead time series prediction strategies that track recent trends by giving more weight to recent data. The error of this algorithm is about 2% and 55% less (36% less on average) compared to Network Weather Service (NWS) system. There are proposed two types of approaches static and tendency-based, but the experiment was done using only the tendency-base since the static prediction strategies always give worse results than does a simple last-value prediction strategy in the initial experiments. It was explained that for TS is impossible to forecast when a time series is going to "change direction" – that is, when an increasing time series will become a decreasing one, or vice versa [20]. At this turning point is where the highest error occurs. For these tendency-based algorithms for example if a point drops after several increments the predicted values will still increment by the reason of the tendency of the TS. He discovered that all experiments had lower error than NWS and for lower frequencies, the reason for it may be that data points are widely spaced in time.

The case of Liang et al. [3] is a proposition of a model with a longer prediction for CPU Load range, with an acceptable accuracy compared with its previous predictors. For this algorithm was used Fourier Transformation (FT) to exploit the periods of the CPU waves. All the experiments were done by comparing the tendency-based algorithm proposed by Yang et al. [20], the same algorithms were tested using one step ahead prediction. Using MM, MER and SD-MER, the results showed that the long term prediction has a lower error rate compared to the short term and higher accuracy in most cases compared to the tendency based algorithm. On the other hand, the mean error at each point should be lower even though it had high precision.

As it was shown in the previous study LSTM was the best algorithm predicting CPU resource. Nääs Starberg and Rooth [2] predicted CPU utilization using the same algorithm compared to MA as a naive baseline model and standard RNN using MSE and MAE as evaluation metric. The second question was about how

far in the future can predict CPU with acceptable accuracy. As expected LSTM is an improvement of the standard RNN, even for a multi-step ahead prediction which were for 10, 20 and 30 steps ahead. However it is clear that the further away we forecast the higher the error gets.

Rahmanian et al. [19] proposed an ensemble prediction algorithm that combines several prediction models to increase profit by gathering more customers and providing efficiency in cloud resource usage. Also, it is mentioned that there is no single model that can accurately predict the future usage of different cloud resources. The ensemble proposed has a weight that is penalized if the algorithm has no good performance or the ones with better accuracy get awarded. In each time slot, each prediction model is evaluated to determine its weight, after the update the ensemble prediction results on combining the multiplication of the predicted values of individual prediction models to their weights. Rahmanian et al. [19] proved that this type of ensemble outperforms the references.

Baig et al. [18] proposes a method to automatically identify which model predicts more accurately based on the statistical feature of historical resources usage. The historical resource is divided into slices of size k, each slice is used to train different prediction models then the system selects the model with lower prediction error for a given slice and keeps the selected features for the best predictor. Baig et al. [18] uses a library TSFRESH to filter features, removing features with identical unique values, highly correlated with one another and the ones that have zero importance for a given set of features, some of these features are standard deviation, partial, and autocorrelation at different lags and the first location of minimum. MAE and RMSE is used to select best model and RF outperforms all predicted models. Baig et al. [18] got an improvement of 6% to 27% for prediction accuracy compared with the references and there is also a remark on the window sensitivity.

Baig and de Catalunya. Departament d'Arquitectura de Computadors [17] proposes three approaches to optimizing services, first techniques to downsize the database yet without losing relevant information. The second solution proposes an adaptive prediction model, this method dynamically selects the best prediction model for estimating and forecasting cloud resources based on time series features. This method was implemented because Baig and de Catalunya. Departament d'Arquitectura de Computadors [17] observed that different predictors yield better estimation for different scenarios of forecasting. Some methods used were LR, SVM, Gradient Boosting Tree (GBT) and Gaussian Process also known by Krigin (KR). Using RMSE and MAE Baig and de Catalunya. Departament d'Arquitectura de Computadors [17] proved that RF had better performance, he also proved window size sensitivity for this solution. The third solution proposed was a method to dynamically identify the best sliding window size to train the regression model for estimating and forecasting cloud resource utilization. Baig and de Catalunya. Departament d'Arquitectura de Computadors [17] concluded that the third approach requires an extensive search to identify appropriate window size which is not feasible for real-time processing.

Another different form of applying an automation is the forecasting is proposed by Wagner and Michalewicz [7], where we encounter an adaptive windowing for

time series forecasting in dynamic environments. It compares GP with Dynamic Forecasting Genetic Program (DyFor GP) that turns out to behave better by more than 50%. The most used methods to forecast time series are applied in a static environment and require some element of human judgment at the beginning and therefore are subject to error. In the real world situation environments are continuously shifting situations, of course it depends of the area. Still, these environments ask for adaptive methods and meet these changing conditions. This topic is approached by Wagner and Michalewicz [7] where GP is compared to DyFor GP which is an adaptive windowing technique that automatically adjust to variations over time. Here the sliding window approach is explained as in the scheme Figure 4.3a on section 4.3. This feature allows DyFor GP yo analyze all existing data and take advantage of previous patterns making it easier to adapt and forecast. It is also studied the correct size of the training data automatically. During this experiments it is only forecasted the 1 value ahead and DyFor GP performs well over all dataset with difference over 50% compared to the GP.

Rao et al. [12] performs a time series forecasting of CPU utilization using, ARIMA. Triple Exponential Smoothing or Holt-Winters (TES) and LSTM in order to compare them and conclude which one is more relevant using MAPE and MAE as metric. Mahalakshmi et al. [1] proved that LSTM has better performance as the results had lower MAPE, ARIMA was the next and the worst performance was given by TES. The explanation for these results was that the dataset used was unlike the other as it had a lot of residual error which is not suited for time series prediction algorithms.

Although for the case of Rao et al. [12] TES wasn't a good forecasting algorithms, but as explained that happened because of the high number of errors, there are cases were TES performed as the best algorithm as is the case of Banitalebi et al. [21] , Santos and Pour Yousefian Barfeh [29] or Karaman and Altiok [22] where it was testes in simple cases with 1 step ahead or with simpler algorithms like ARIMA.

The case of Akrami et al. [23] there were tested two algorithms with a higher degree of complexity like TES and SARIMA with multi-step ahead where the evaluation metric used was RMSE. In order to identify the parameters of the algorithms the Box–Jenkins method was used, this methods was explained on section 2.3. The experiment was done by forecasting with 1, 3,6, 9 and 12 months using SARIMA and TES, where SARIMA performed better in this case, even though the accuracy of the model decreases when the lenght of step ahead increases.

We have a last study were Idman et al. [24] modeled a time series with AR, ARMA, SARIMA, Double Exponential Smoothing (DES) and TES usign RMSE as evaluation metric. Explained that time series is a combination of various components:

- Trend, long term behaviour over time
- Seasonality, repeat period over time
- Error (Residual), unexpected change of data over time

If trend and seasonality component don't change over time then an additive method should be used:

$$Y = T + S + E \tag{2.14}$$

If trend and seasonality increase or decrease non-linearly over time, then a multiplicative method should be used:

$$Y = T * S * E \tag{2.15}$$

For this study the methods showed before were used to predict from 1 to 11 steps ahead, or months ahead. Unlike before, for more than 1 step, the error decreased and then kept almost constant after 5 steps. This happened because the point when the test began was when the tendency changed, so after the first step the error decreased because of mean of values.

2.5 Summary

This chapter explained the work and data context, from where data was extracted, and what kind of data we can expect. It exhibited the two types of data we have, business and operational, and its charging process. It is known that the operational site consists of three sites, each with its number of machines. Therefore its dataset is gathered into a single for each site with the overall information per site. All features are merged into a single business data for business data.

Our problem consists of receiving multiple tabular time series from where there is a need to understand its peak tendencies or their tendency to change and how it affects the process. It presented the research done before applying techniques to data and how to extract valuable information from forecasting. For example, suppose we have too many features which can mislead forecasting results. In that case, we may be able to apply some techniques of feature importance and selection or, if we have missing data, how to handle or categorize them.

After the research, it was elucidated that we have two types of forecasting, as it can be done using machine learning or traditional time series. Depending on the forecasting method, overfitting or underfitting can be avoided, as well as preliminary work, time, or memory.

The related work showed several implementations in this area, each with its techniques and metrics. It can be seen the summarized information about forecasting in Table 2.5 or about feature selection in Table 2.4. The best performance technique when forecasting using machine learning was LSTM and for traditional time series TES and SARIMA because of its ability to use time series statistics to perform long-time forecasting. In the case of this dissertation, the evaluation metric which fits better is MSE because we want to penalize peaks. However, MAPE is used as an additional metric to get a better interpretation.

Chapter 3

The Approach

This chapter will present how to approach this dissertation's goal by considering the dataset, its context, related works, and the best models suited for this dissertation.

It will first present use cases when interacting with the tool, the requirement specification of the solution, and how the solution will be delivered. Then, at the end of the chapter, the pre-process and some visualizations of its results.

The main goal is to forecast resources from the data that AlticeLabs provides. In the section 2.1 are presented the dataset in detail. There is a need for monitoring and forecast resources to support the solution for real-time control and charging of telecommunications services, called NGIN - Next Generation Intelligent Network. This optimization aims for a better service for the client and improved use of resources in an automated form.

The data created by the company can be categorized into two main areas:

- **Business**, produced by the business application.
- **Operational**, produced by the operational application.

The business intelligence application produces reports from the activity information of the business processes section 2.1.

The operational management application produces operational reports that are originated by the operational Key Performance Indicatorss (KPIs) generated periodically by the application instances.

To achieve the expected goal, we can take advantage of the historical data or the multi-feature dataset to find patterns and forecast the values of the next day. In the end, correlate the business and operational data with the help of a graphical tool.

We have a dataset that has been withdrawn from the stations and given to us. Now there is a need to automate the process of receiving data, pre-processing, and storing them in a database. At this step, we will study them by first visualizing them, understanding the problem, choosing some possible outputs to predict,

and testing the model with some algorithms that fit better for each case. Also, we need to understand what is the best approach to apply. Chang and Tsai [28] explained that there are two types of approaches, to use Time Series (TS) methods or the associative model. TS models assume the future forecasting value is related to the historical pattern. On the other hand, associative models assume the predicting value correlates with the variables considered in the model.

3.1 Use Cases

The knowledge that can be gathered so far can be used to construct a use case to justify the framework decision described in section 3.3. Use cases can be presented in a list or event steps. Exhibits interactions between a role and a system are typically defined to achieve a goal. It also can be a set of behaviors that, in this case, the tool may perform in interaction with its users. This interaction produced observable results that contribute to the user's goals. The use cases in the context of this work having in mind operational and business data, are represented in Table 3.1

Use Cases		
Nr.	Action	Tool Interaction
1	Visualize how business data affects CPU Load.	The application displays the CPU Load forecasting and business features.
2	Compare the forecasting error of a day to the whole forecasting.	The application allows the option to visualize the forecasting error of a time range or the whole forecasting.
3	Visualize all forecasting models for all sites for a specific machine.	The application allows choosing a machine for each site CPN, SMP, and DSGW to display a graph containing all the forecasting models.
4	Compare the percentage of successful and error requests for business data.	The application allows the user to compare the percentage of successful requests with the percentage of requests that resulted in an error.

Table 3.1: Table of Use Cases

Prediction of peaks is a challenging task, and some can even say that it is impossible [20], as a result of being in a changing environment. However, we can be prepared for it, and forecasting with relatively high precision can lead to business improvement.

In Table 3.1 is exposed some possible scenarios when using the tool. Most of the cases are focused on forecasting models and how to correlate peaks with business data. The use cases shown in the are based on some visualizations that will be described during this chapter.

3.2 Requirement Specification

MoSCoW Prioritization, also known as MoSCoW Analysis, is used for the requirement specification the tool used to support and map the demands was MoSCoW Method. It is a prioritization tool used in management to agree with stakeholders on the importance of the deliverable. MoSCoW is an acronym for Must Have, Should Have, Could Have, and Won't Have.

In a spell, out way, these criteria can be evaluated by the importance of a company. Must have is critical, should have is important but not necessary, could have is desirable but not necessary, and won't have was decided between stakeholders as the least critical.

The requirements covered in the table Table 3.2 were determined within a collaborative approach. Their establishment and categorization were performed fortnightly during meetings of the first semester, from whence they resulted in an agreement.

MoSCoW Table			
Must Have	Should Have	Could Have	Won't Have
Dashboards with forecasting and business data in Grafana	The solution should be automated	Real Time Analyses Execution	Cyber Security implementation
The solution must be able to be imported under Next Generation Intelligent Network (NGIN)	Train forecasting models using multiple features	Adjustable forecasting model type	
One day ahead prediction	Connection between Grafana and database	Adjustable Window Size Prediction	

Table 3.2: MoSCoW Table of the Requirements of the Application

The central and most fundamental requirements for this dissertation to be completed are in the "Must Have" and "Should Have" columns.

3.3 Framework

The first step of the proposed solution in Figure 3.1 is by pulling the raw data and pre-processing them. This is an automated process since the goal is to standardize the datasets and prepare them for the forecasting system. The chosen algorithm within the forecasting system can be composed of a single forecasting model or several depending on their performance on the overall error or on outliers, as will be seen in the next chapter. In this forecasting system, all the models already have pre-defined parameters, and they are trained with the new samples. The one day ahead forecasting result is stored in a database that is linked to Grafana. This tool is an open-source web application for interactive visualization. Dashboards created within the tool can be exported or imported in .json format, allowing integration within NGIN framework.

An internal plug-in link the pre-defined dashboard to the target database, where forecasting and pre-processed data are stored. Using this plug-in automatically assembles the results of the current forecasted models with the historical ones in a visualization form in Grafana.

The stakeholders chose and approved these visualizations as they met the company's requirements.

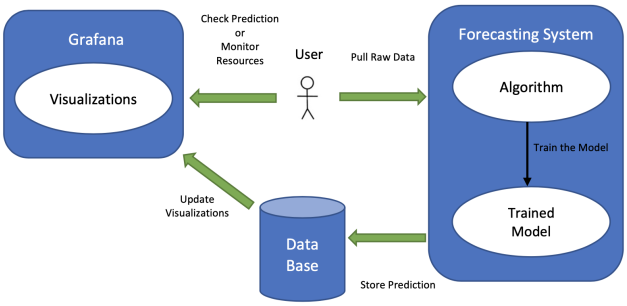


Figure 3.1: Architecture Flow of the Proposed Application

In the scheme presented in Figure 3.1 we see that there are three components and a user. This user has two associated actions with two types of contributions, one to help the forecasting system and the other as a final user observing the final output of the forecasting system. The scheme represents an example of one or multiple users, one to check the visualizations and one that gives the instructions to start the process of forecasting data.

When integrated into the working environment, this architecture will not need direct human interaction to start the forecasting process, as it can be set automatically daily.

3.4 Pre-Processing

Our dataset is updated every month, where within, we have metrics from periods of one hour every day. This dissertation's goal is to identify patterns and therefore forecast problems that may occur. An exhausting study of the dataset is needed to understand the dependencies between data and their patterns.

The first step is looking for missing data, noisy data, or semantics. There was human involvement in extracting data from a higher aggregation of data, which might be a higher error percentage.

After checking the content of it, there is a need to rearrange them to visualize or have a better train performance. After the dataset definition in section 2.1 where it was described that there are two types of data, the business which is composed of five features, and the operational data composed of three different sites, satellite, central, and Service Management Platforms (SMP).

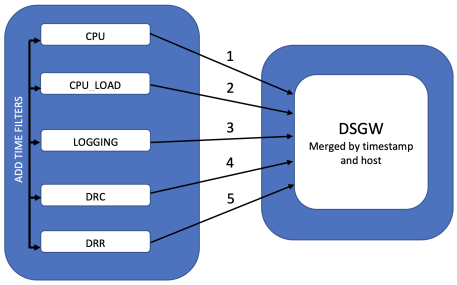
3.4.1 Data Pre-processing

Requests are received at Satellite Site (will be referred to as Diameter Gy (DSGW)). Here is the only site where request types can be differentiated. DSGW sends a charging request to the Central site (will be referred to as Central Processing Node (CPN)), here charging requests are executed, and finally, SMP will keep a record of the events.

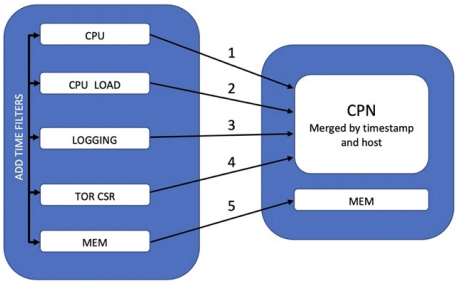
Each site has its scheme since the information extracted differs. One process that was maintained between them is as illustrated, "ADD TIME FILTERS". With this filter, a column is created for the day of the week and another column for the hour of the day. These two columns are added for all of the datasets. The additional features will help discover patterns by filtering out a day of the week and hour of the day and can also be seen as additional features.

Per each scheme, arrows represent data selected from the dataset, and the meaning of each arrow is explained below:

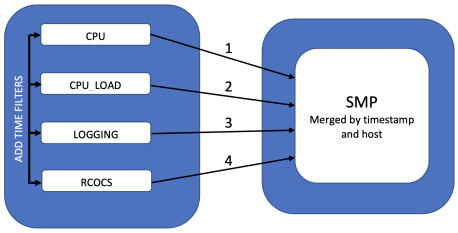
- **DSGW**, pre-process is shown in Figure 3.2a
    1. Minimum, maximum and average
    2. Minimum, maximum and average
    3. Minimum, maximum, average, and sum per metric.
    4. Minimum, maximum, average, and sum per KPI and metric.
    5. Minimum, maximum, average per KPI
- **CPN**, pre-process is shown in Figure 3.2b
    1. Minimum, maximum and average
    2. Minimum, maximum and average



(a) Scheme of DSGW Pre-Process



(b) Scheme of CPN Pre-Process



(c) Scheme of SMP Pre-Process

Figure 3.2: All sites Pre-process Scheme

3. Minimum, maximum, average, and sum per metric.
4. Minimum, maximum, average, and sum per KPI and metric.
5. Minimum, maximum, average per metric.

- **SMP**, pre-process is shown in Figure 3.2c

1. Minimum, maximum and average
2. Minimum, maximum and average
3. Minimum, maximum, average, and sum per metric.
4. Minimum, maximum, average, and sum per KPI and metric.

The result in 4 datasets one per site, DSGW, CPN, SMP domain, and an extra one for CPN. The result of these datasets was obtained by its inner aggregation of timestamp and host.

In order to make it easier to interpret, the names of the host per site were mapped into integer numbers from 1 until the number of machines per site. It can be seen in the tables 3.3,3.4 and 3.5.

When looking at data apparently doesn't present any "NaN" but there are missing values as it is presented in the tables 3.6, 3.7 and 3.8.

In our case, missing data is a problem because nearly all standard methods presume no missing data, in other words, complete information for all variables included in the analyses as said in Soley-Bori [30].

We can see in the table 3.6 that all machines have 11 missing hours, and it should be noted that it is the same for all of them.

Name Machine Mapping (CPN)	
Number Machine	Hostname
1	odosdprd-pcco2cs-node01
2	odosdprd-pcco2cs-node02
3	odosdprd-pcco2cs-node03
4	odosdprd-pcco2cs-node04
5	odosdprd-pcco2cs-node05
6	odosdprd-pcco2cs-node06
7	odosdprd-pcco2cs-node07
8	odosdprd-pcco2cs-node08
9	odostprd-pcco2cs-node01
10	odostprd-pcco2cs-node02
11	odostprd-pcco2cs-node03
12	odostprd-pcco2cs-node04
13	odostprd-pcco2cs-node05
14	odostprd-pcco2cs-node06
15	odostprd-pcco2cs-node07
16	odostprd-pcco2cs-node08

Table 3.3: Mapping the hostname to the representing number for each machine (CPN)

Name Machine Mapping (DSGW)	
Number Machine	Hostname
1	odosdprd-pcco2cs-dscp1
2	odosdprd-pcco2cs-dscp2
3	odostprd-pcco2cs-dscp1
4	odostprd-pcco2cs-dscp2

Table 3.4: Mapping the hostname to the representing number for each machine (DSGW)

Name Machine Mapping (SMP)	
Number Machine	Hostname
1	odosdprd-pccsmp-be1
2	odosdprd-pccsmp-be2
3	odosdprd-pccsmp-be3
4	odostprd-pccsmp-be1
5	odostprd-pccsmp-be2
6	odostprd-pccsmp-be3

Table 3.5: Mapping the hostname to the representing number for each machine (SMP)

Real Data Statistics per Machine (DSGW)				
Machine	Min	Max	Mean	Missing Values
1	50.92	698.25	197.52	$11 \approx 0.32\%$
2	78.38	737.28	212.8	$11 \approx 0.32\%$
3	36.57	523.18	201.71	$11 \approx 0.32\%$
4	42.67	547.68	202.68	$11 \approx 0.32\%$

Table 3.6: Minimum, maximum and mean of the real data per Machine DSGW: higher values colored in red and lower values colored in green each statistic

Real Data Statistics per Machine (CPN)				
Machine	Min	Max	Mean	Missing Values
1	97.87	1602.65	402.13	0
2	297.07	2198.32	625.52	0
3	213.23	1997.18	521.62	0
4	203.25	1828.25	516.82	0
5	224.27	1939.05	505.05	0
6	213.93	2154.7	511.28	0
7	206.0	1621.63	503.85	0
8	206.53	1746.58	515.01	0
9	98.8	2343.95	509.59	$1 \approx 0.03\%$
10	117.6	1507.12	509.1	$1 \approx 0.03\%$
11	193.2	1761.47	491.12	$1 \approx 0.03\%$
12	97.8	1599.55	490.24	$52 \approx 1.49\%$
13	85.4	1595.25	495.83	$52 \approx 1.49\%$
14	72.8	1873.1	502.84	$52 \approx 1.49\%$
15	137.74	1572.65	480.94	$52 \approx 1.49\%$
16	168.03	1644.3	489.01	$52 \approx 1.49\%$

Table 3.7: Minimum, maximum and mean of the real data per Machine CPN: higher values colored in red and lower values colored in green each statistic

Real Data Statistics per Machine (SMP)				
Machine	Min	Max	Mean	Missing Values
1	76.17	422.98	225.61	0
2	79.82	424.77	233.65	0
3	65.78	431.03	222.83	0
4	39.08	516.65	239.58	0
5	49.43	499.6	248.75	0
6	45.72	514.52	237.93	0

Table 3.8: Minimum, maximum and mean of the real data per Machine SMP: higher values colored in red and lower values colored in green each statistic

When we look at table 3.7 that there are eight machines with missing values, where machines 9, 10 and 11 have one missing hour and from 12 to 16 have 52 missing hours.

Looking at the SMP in table 3.8, we see no missing values.

A study about the characterization of missing data was discussed on section 2.3, and techniques overcome this issue and adapt data to the model in section 2.3. Having in mind data studied in section 2.1, it can be said that the best way to handle missing values is by using Last Observation Carried Forward (LOCF) but with a slight difference. Instead of using exactly the last one, because our data has a seasonality of 24h, as we will see further in this dissertation will be more appropriate to use the value from the last day at the hour of the missing value. As can be seen in Figure 3.3, data may be in the presence of peaks on the previous values, meaning that when using LOCF, unnecessary peaks may be added. To overcome this issue, two techniques may be mixed, LOCF and the Mean Substitution value of the hour of the peak of the historical data, as we can see in Figure 3.3

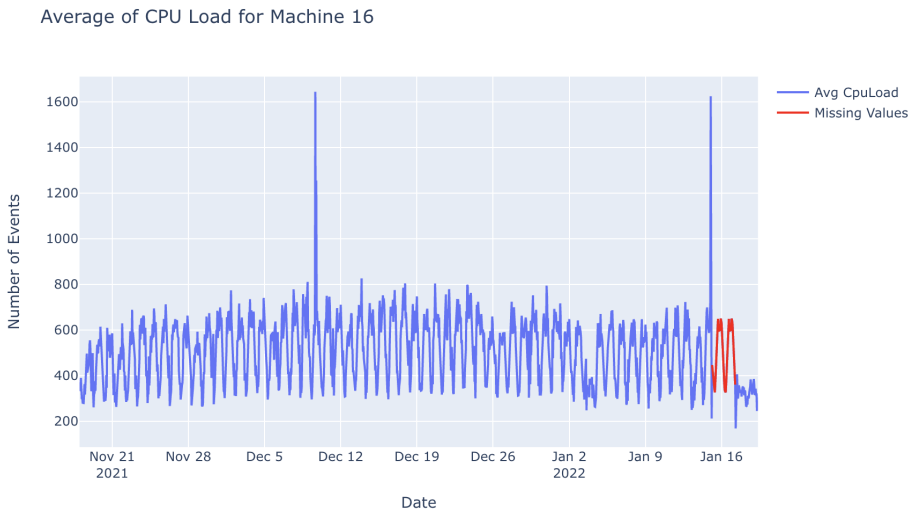


Figure 3.3: Pre process of Missing Values for Machine 16 (DSGW)

In the case of integrating into the NGINPCC, it is done automatically and saved into a separate database to be linked to Grafana and the forecasting used.

In the case of business pre-processing, it is an easier task. We take the five fea-

tures, and then we mix them by merging on the date column. An extra column is created for future visualization or to make the combination easier.

3.4.2 Raw Data Visualizations

Visualizations are a graphical representation of the information and, in this case, temporal, tabular data. Visual elements like Scatter Plot, Charts, and Graphs make it easier to understand data, outliers, and patterns.

In this sub-section, raw data visualization from both operational and business data is analyzed. Its high or low peaks and patterns in common between machines or sites.

First, it starts by analyzing features per machine from the operational data and comparing a slice's peaks and patterns with the entire dataset to see if patterns and peaks maintain. Then at the end of this sub-section, the business data behavior and peaks are analyzed.

Operational Data

It starts with analyzing the successful events of each type of request, voice call, SMS, or data session with the interest to see how it behaves over time and compare them to the total number of events per request. To do that, the percentage error of the request is calculated using Equation 2.8

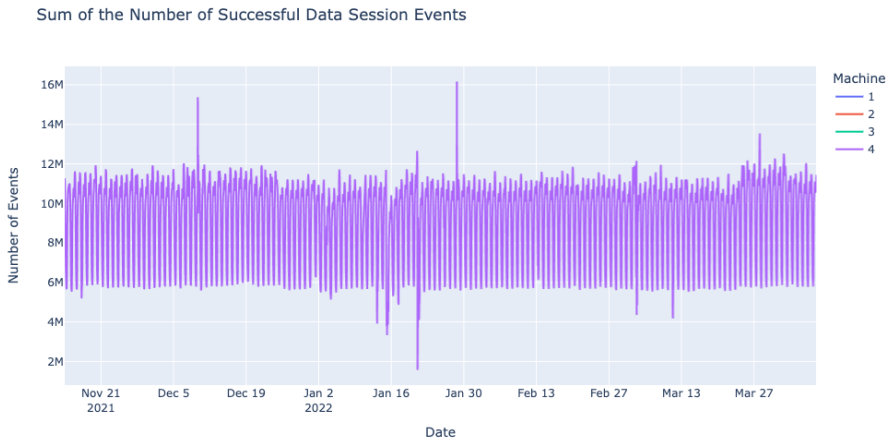


Figure 3.4: Entire Dataset of Successful Data Session Events per Hour for each Machine at DSGW site - The machine's number corresponds to the DSGW host mapping

When looking at successful data session events, Figure 3.5 over time, we can see that the decay starts around 11 PM, reaching the lowest values around 8 AM. We can also see that some smaller decays start at 4 PM and reaches the lowest at 7 PM. We can see a clear pattern during the day that repeats daily. However, it is not clear when we look at the big picture on Figure 3.4. Some peaks can be

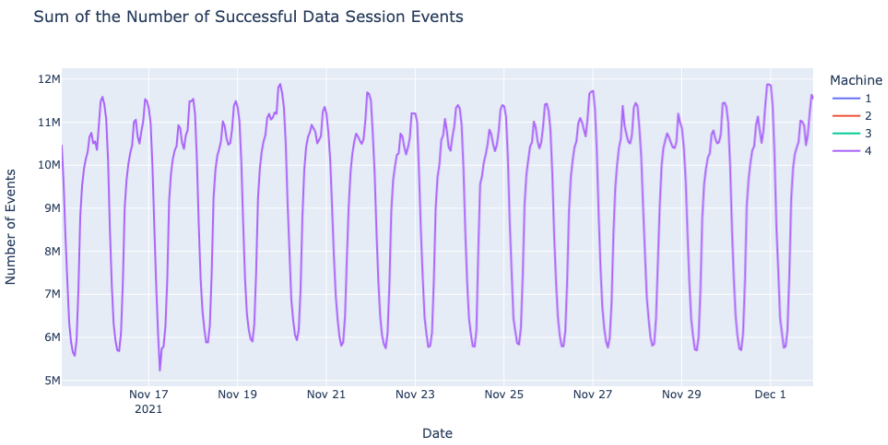


Figure 3.5: 15 days of Successful Data Session Events per Hour for each Machine at DSGW site - The machine's number corresponds to the DSGW host mapping

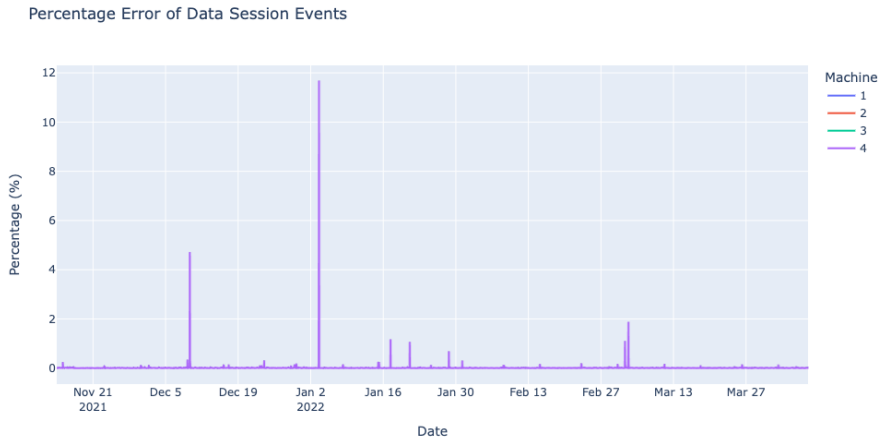


Figure 3.6: Entire Dataset of Percentage Error of Data Session Events per Hour for each Machine at DSGW site - The machine's number corresponds to the DSGW host mapping

seen but do not present any pattern as they happen during any day of the week or month. When looking at Figure 3.6, data session request which resulted in an error, can be seen that the highest error was on 3rd January at 14h with 12% of error, but there is no pattern between these peaks.

In the case of the voice calls, Figure 3.8, the first thing that pops up is that there is one day with lower values than others, and the second is the difference between the number of Data Session Events and the Voice Call events. The higher value for data sessions is around midnight and for the voice calls is around 2 PM or 3 PM. The day when there are fewer voice calls is Sunday, and we can see this pattern is repeated for the entire dataset in figure Figure 3.7. After 2 PM or 3 PM, the requests decrease until the second lowest value of the graph at 5 PM. Again it increases a little bit, arriving at the second-highest value in a day at 9 PM and then decreasing to the lowest value at 7 AM. When we look at the percentage of error of voice call request Figure 3.9, we can see that it is even higher than the

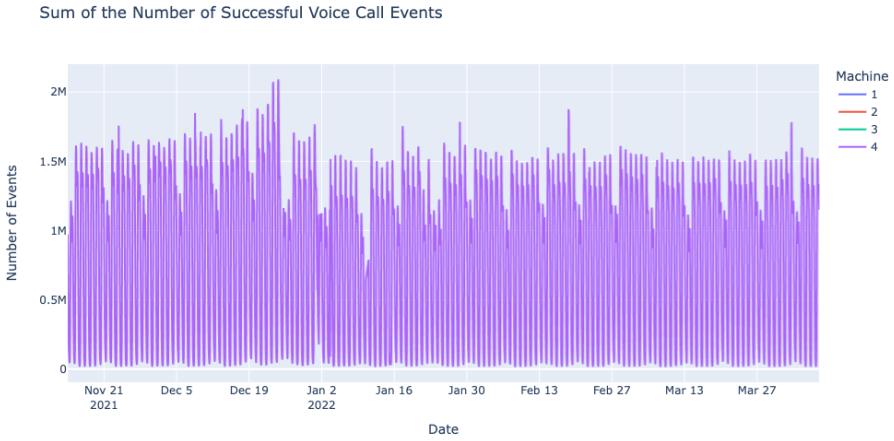


Figure 3.7: Entire Dataset of Successful Voice Call Events per Hour for each Machine at DSGW site - The machine's number corresponds to the DSGW host mapping

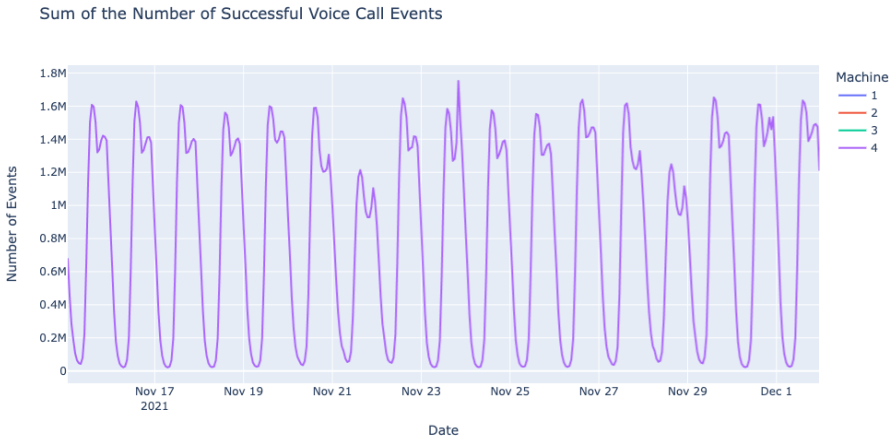


Figure 3.8: 15 days of Successful Voice Call Events per Hour for each Machine at DSGW site - The machine's number corresponds to the DSGW host mapping

previous, reaching 33% of error.

Successful SMS events can be seen looking at Figure 3.11, there are even fewer requests, as the highest is around 20K. It is not easy to see which hour of the day the highest value is reached. We know that it can be at 4 PM or 11 PM. Although we can see that the lowest of the day is at 8 AM, the second-lowest is at 7 PM, and the second highest is at 3 PM or 4 PM. When we look at the entire dataset on figure Figure 3.10 we see a change in the pattern from 21 December to 22 December. Even so, when we look at the figure Figure 3.12 where we can see the percentage error, we can conclude that the total number of SMS requests also changed. In the same graph, the percentage error is still high compared to the data session, but it is the same as for voice calls.

As expected, there are fewer requests during the night than during the day. Also, the usage of data sessions being higher than the others was expected as well, as

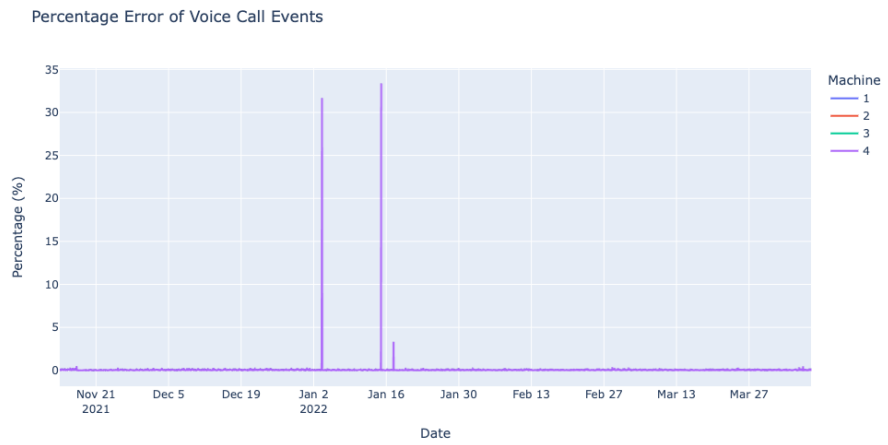


Figure 3.9: Entire Dataset of Percentage Error of Voice Call Events All Data per Hour for each Machine at DSGW site - The machine's number corresponds to the DSGW host mapping

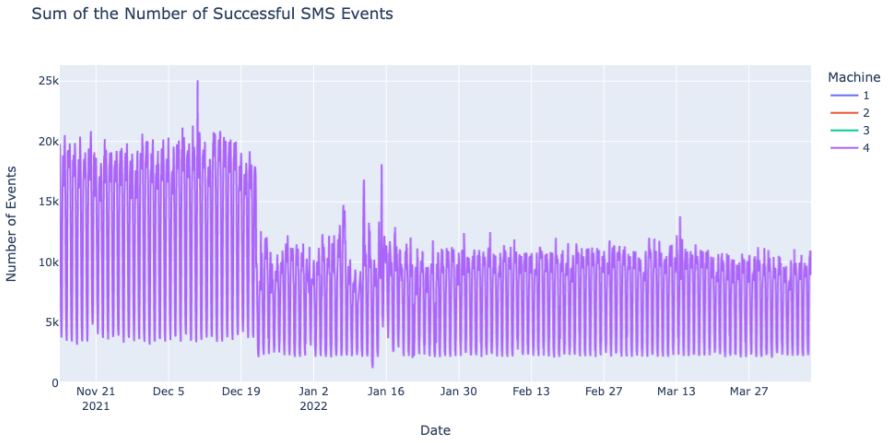


Figure 3.10: Entire dataset of Successful SMS Events per Hour for each Machine at DSGW site - The machine's number corresponds to the DSGW host mapping

it is influenced by marketing to use apps free data session credit charge.

To complement these requests, we can look at the Average percentage of CPUs not actively being used. We see that on January 13th, there is a significant difference between machines 3 and 4 compared to 1 and 2. While during this day, machines 3 and 4 reached the highest peaks, machines 1 and 2 reached the lowest ones. With our data, we do not have an explanation for only these requests, but we can surely tell that the accounts linked to 1 and 2 are composed of a pack of higher accounts than those for 3 and 4. As we studied on section 2.1 DSGW is the site that starts by receiving the requests, and these requests are linked randomly to a machine by the sold packet. This packet can contain a different number of subscriptions as asked in accordance to the company, and further on, this packet is linked to a machine. The discrepancy usually happens when these subscriptions are uploaded into the system, which can create this disparity.

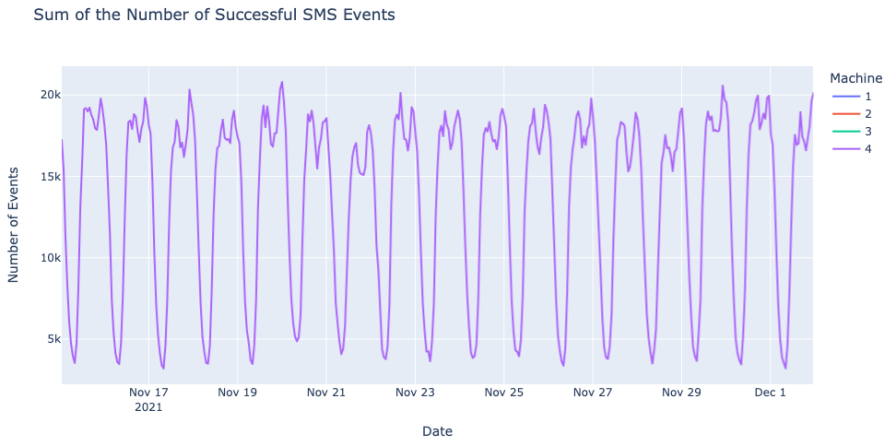


Figure 3.11: 15 dats of Successful SMS Events Sliced Data per Hour for each Machine at DSGW site - The machine's number corresponds to the DSGW host mapping

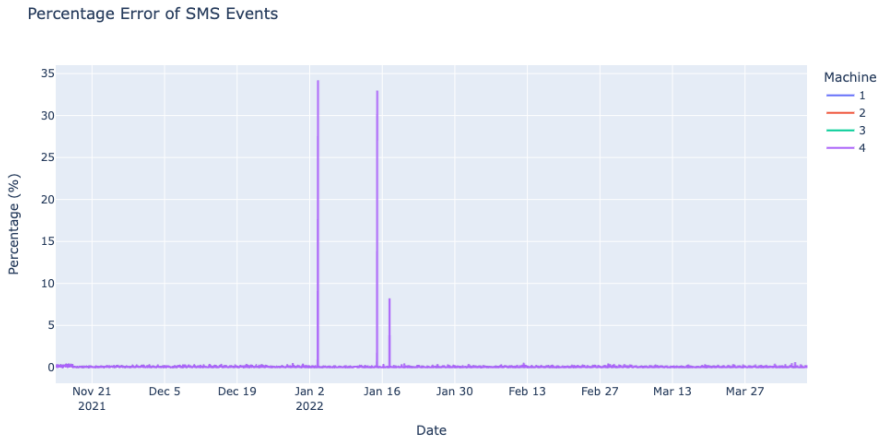


Figure 3.12: Entire dataset of Percentage Error of SMS Events All Data per Hour for each Machine at DSGW site - The machine's number corresponds to the DSGW host mapping

We can see in Figure 3.14 that the lowest points decrease a little for three days, which are Sundays, which means that there are fewer processes for the CPU to work on. That is the exact behavior of voice calls. On Sundays, they decrease. Maybe it indicates that Voice Calls play a significant role when it comes up to the CPU of the DSGW. The hour of the day when CPU is less used is at 8 AM, and the lowest is around 11 PM. This observation makes sense because around 8 AM is when there are fewer requests, and at 11 PM, there are more.

Once more, at the Figure 3.16, we identify a lower maximum on Sundays because of the above explanations. The lowest value is at 8 AM, and the highest is between 2 PM and 4 PM. We can also see that the second lowest is at 8 PM and the second highest is around 10 PM. In Figure 3.15, we can identify some peaks in the graph, but as expected, it complements the CPU not actively being used.

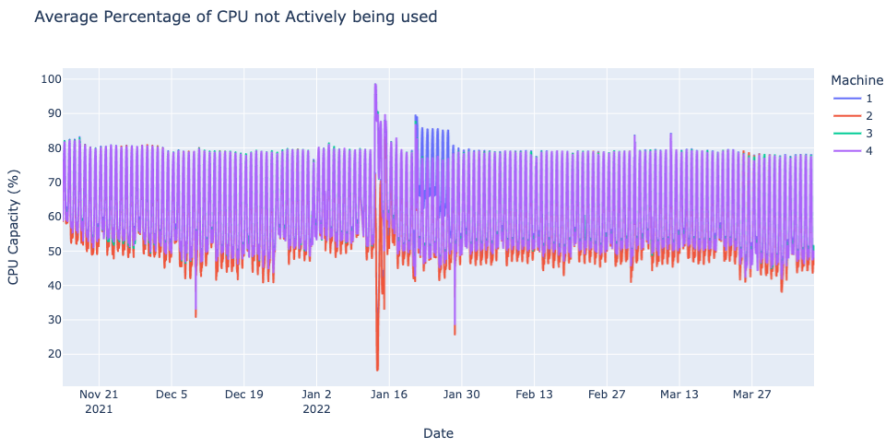


Figure 3.13: Entire Dataset of the Average of the Percentage of Idle per Hour for each Machine at DSGW site - The machine's number corresponds to the DSGW host mapping

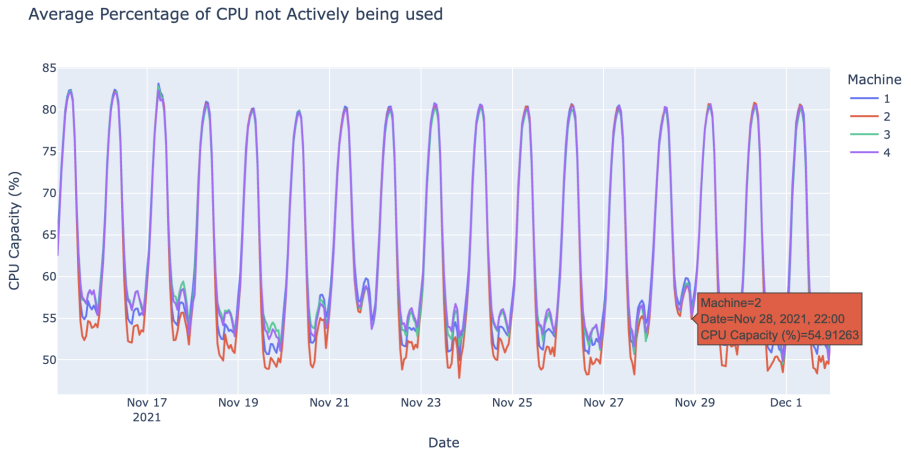


Figure 3.14: 15 days of the Average of CPU Idle per Hour for each Machine at DSGW site - The machine's number corresponds to the DSGW host mapping

At the CPN level at Figure 3.18, there are some strange patterns, until day 17 November the CPU not actively being used was about 57%, after this day it increased for more or less 66%, what they have in common is the hour of the day, which is about 10 PM. Another strange point is what can be identified as an outlier on 25 November at 10 PM. While all the machines had a percentage of CPU not used, around 66%, this machine was about 57%. This can be caused by a human error when extracting the data. We can check the complete data on 3.17 and confirm that there are different patterns over time and no connection between peaks and DSGW.

In the Figure 3.17 we see the opposite behaviour as it was seen in Figure 3.19, as expected, even though with the same strange tendency change. Before 17 November, around 8 AM, can be seen bigger high peaks compared to the ones after 18 November, around 6 AM. Between these days, almost a constant low waiting process per hour per machine can be seen. It also presents a value that looks like an outlier as it is out of the ordinary compared to the other machine's

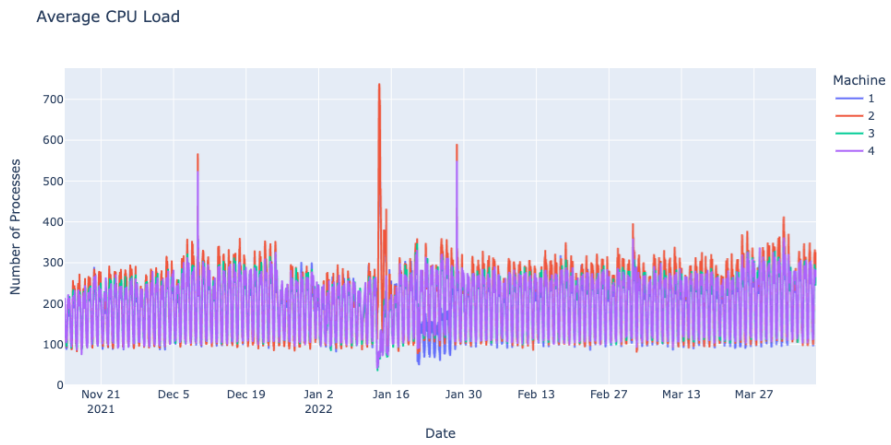


Figure 3.15: Entire dataset of the Average of CPU Load per Hour for each Machine at DSGW site - The machine's number corresponds to the DSGW host mapping

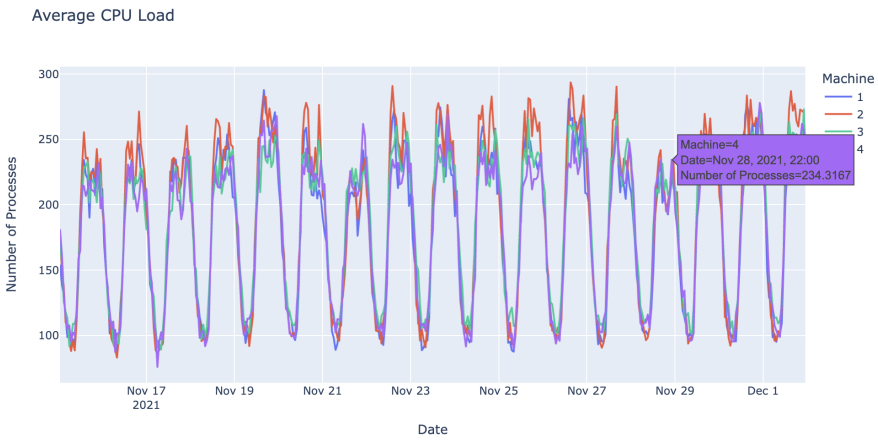


Figure 3.16: Entire dataset of Average CPU Load per Hour for each Machine at DSGW site - The machine's number corresponds to the DSGW host mapping

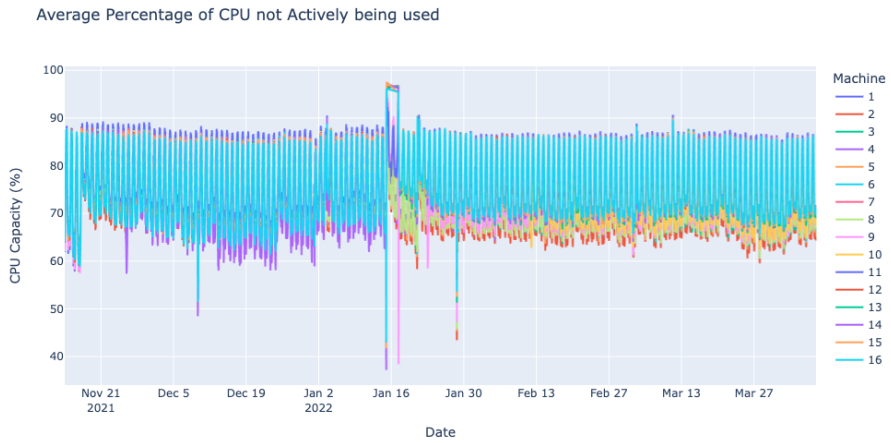


Figure 3.17: Entire Dataset of the Average of CPU Idle per Hour for each Machine at CPN site - The machine's number corresponds to the CPN host mapping

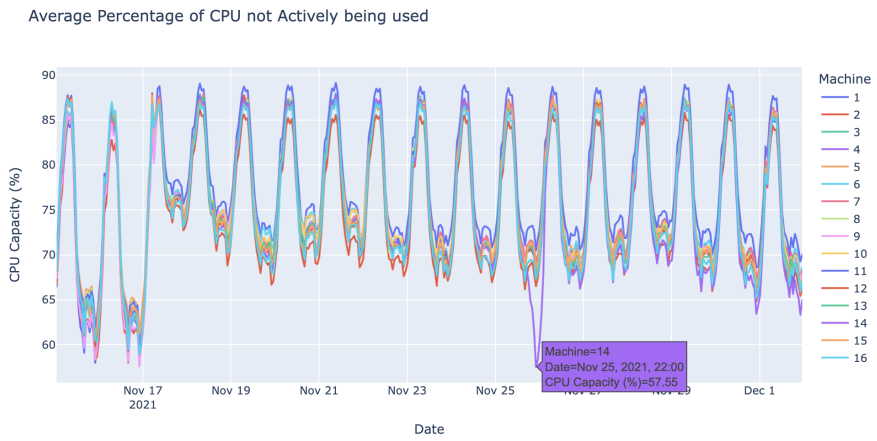


Figure 3.18: 15 days of the Average of CPU Idle per Hour for each Machine at CPN site - The machine's number corresponds to the CPN host mapping

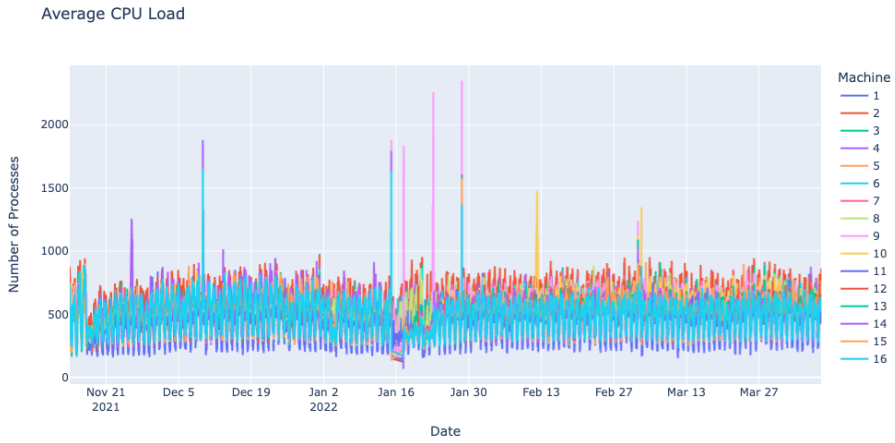


Figure 3.19: Entire Dataset of the Average of CPU Load per Hour for each Machine at CPN site - The machine's number corresponds to the CPN host mapping

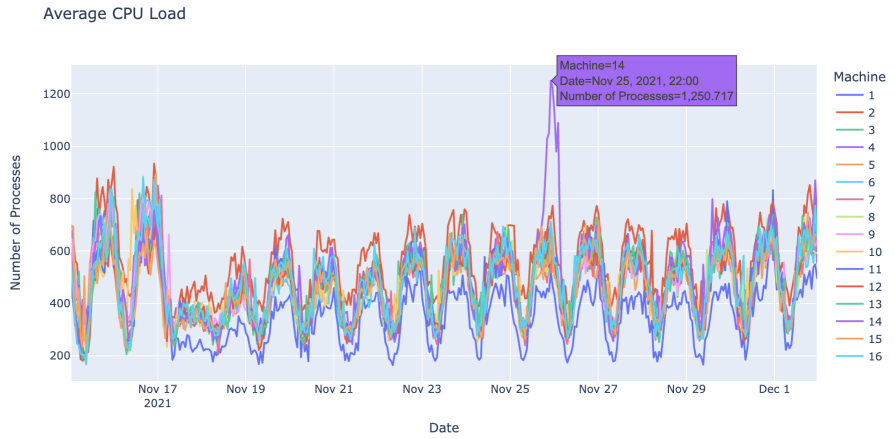


Figure 3.20: 15 days of the Average of CPU Load per Hour for each Machine at CPN site - The machine's number corresponds to the CPN host mapping

high peaks on 25 November at 10 PM.

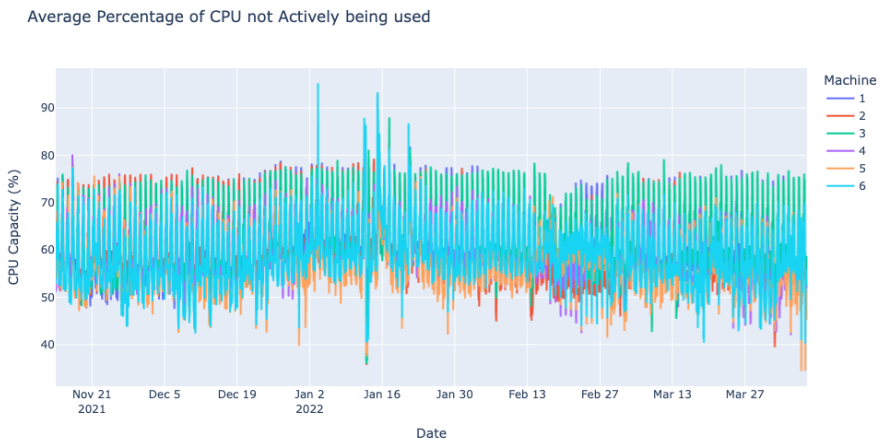


Figure 3.21: Entire Dataset of the Average of CPU Idle (SMP) - The machine's number corresponds to the SMP host mapping

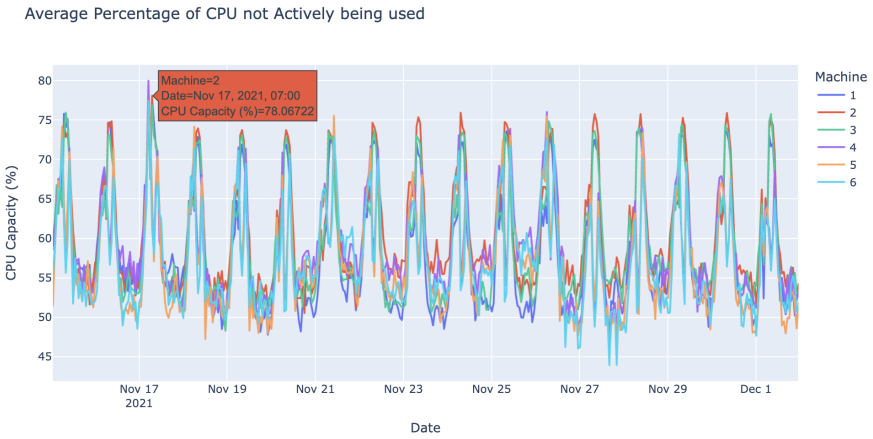


Figure 3.22: 15 days of the Average of CPU Load per Hour for each Machine at SMP site - The machine's number corresponds to the SMP host mapping

When looking at the Figure 3.22, we see that it is constant with a period of a day. The lowest value is between 12 AM and 11 PM. Analyzing the big picture, it does not look that constant. Compared to the CPN, DSGW percentage of CPU not actively being used, the upper bound does not remain as constant, and therefore the waiting processes as well since they complement each other as can be seen in Figure 3.23.

On the Figure 3.24, we see a pretty constant pattern with a period of 1 day. The high is between 4 PM and 10 PM, and the low is between 7 AM and 10 AM. Even though, looking at the big picture, it does not look that constant.

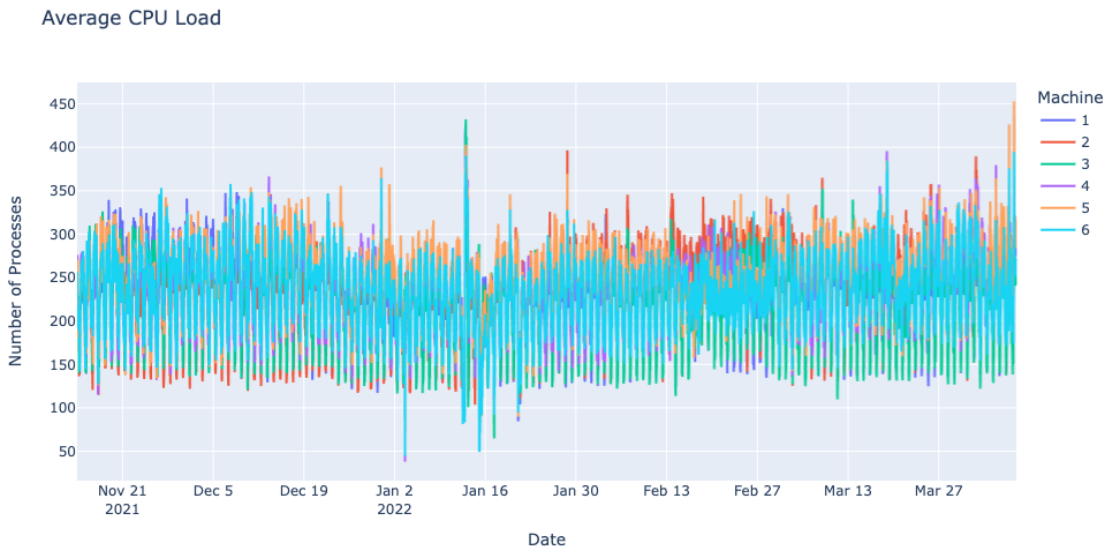


Figure 3.23: Entire Dataset of the Average of CPU Load per Hour for each Machine at SMP site - The machine's number corresponds to the SMP host mapping

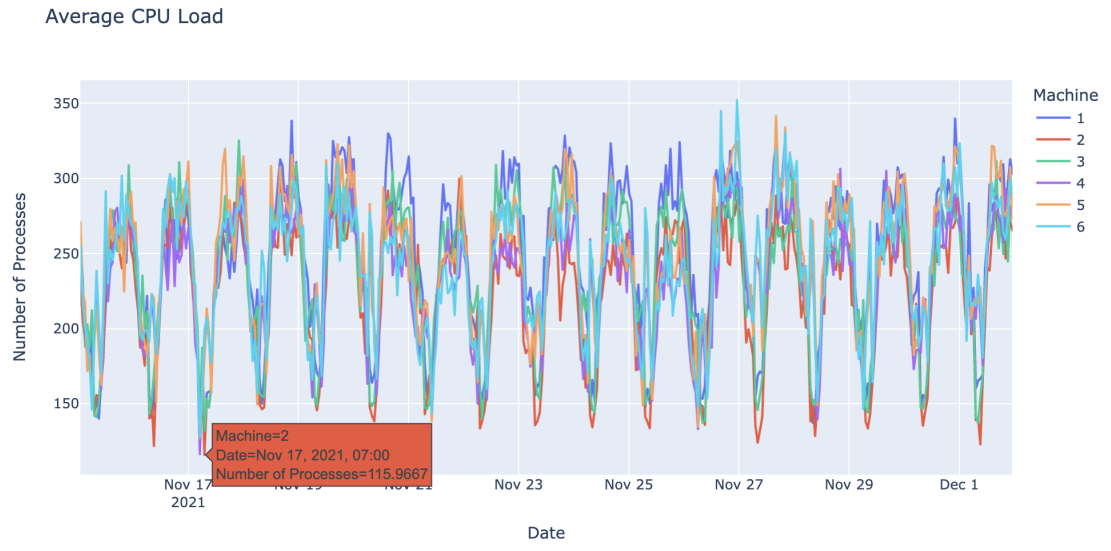


Figure 3.24: 15 days of the Average of CPU Load per Hour for each Machine at SMP site - The machine's number corresponds to the SMP host mapping

Business Data

This sub-sub-section analyzes a dataset with five metrics: SMS, voice call, data session, topup, and provisioning. Each metric is explained in more detail on section 2.1.

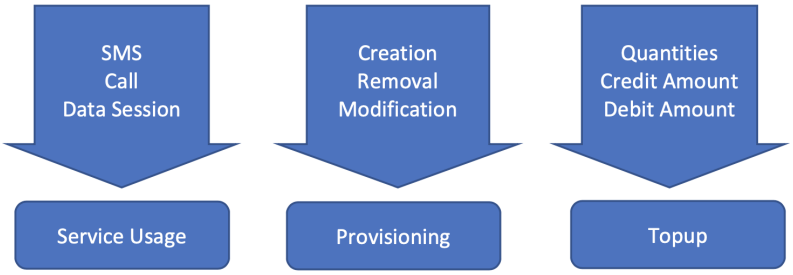


Figure 3.25: Specification of each Business Metric

Let us look at the statistics about each metric from Figure 3.25. First, we see that the quantities differ from one another.

We can see that, as expected, there are fewer topup and provisioning operations compared to the data created by the service usage. The most used service is data sessions which makes complete sense as it is the tool that a person or a company makes the most use of.

Business Statistics			
Metric	Min	Max	Mean
SMS	2370	32316	11211.36
Topups	2	30802	11525.98
Voice Call	7863	884827	354606.01
Data Session	3557050	8699745	6769457.38
Provisioning	855	126292	29830.72

Table 3.9: Statistics for each Business Metric, min, max and mean

Looking at the graph of the Call in figure 3.26 we can see that it has a seasonality of 24*7 hours (one week), meaning that it has the same pattern every week, where at 3 AM we have the lowest values and at 10 AM the highest. The second lowest peak during the day happens at 1 PM, and the second highest peak is at 3 PM, which means that during lunch, people do not call as much as at other times during the day. We can also see a day in the week when the voice call peak does not reach the same values as other days. When looking carefully, we can see that it happens on Sundays. We can also see three days in the graph with low peaks on the 15th, 16th, and 17th of April because of Catholic Easter.

The second graph to look at is SMS on figure 3.27. Although SMS service is not used as much as the previous one, it has a week's seasonality. During the day, the lowest peak is at 3 AM or 4 AM, and the highest is at 12 PM or 2 PM. During the week, we can see that the lowest values are during the weekend. In this case, although the values of Sundays are almost always lower than Saturdays, it is not

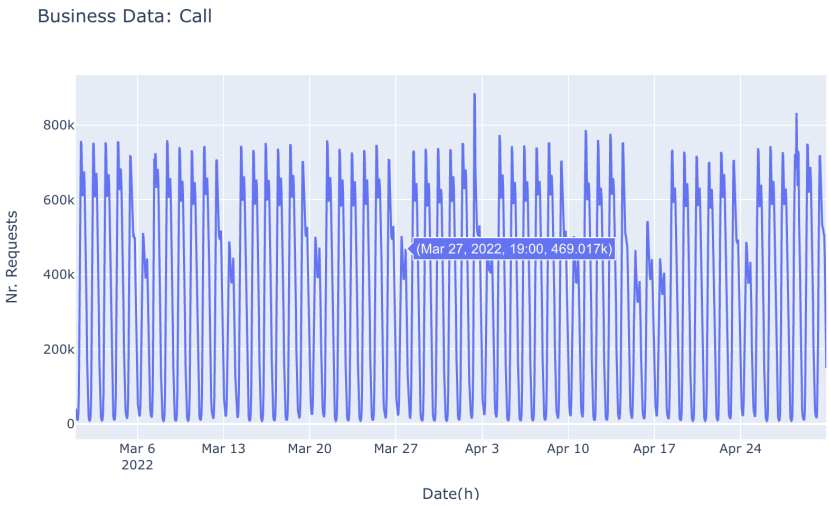


Figure 3.26: Business: Voice Call

that obvious. The highest peaks in the graph are at the end of working days, Thursdays and Fridays.

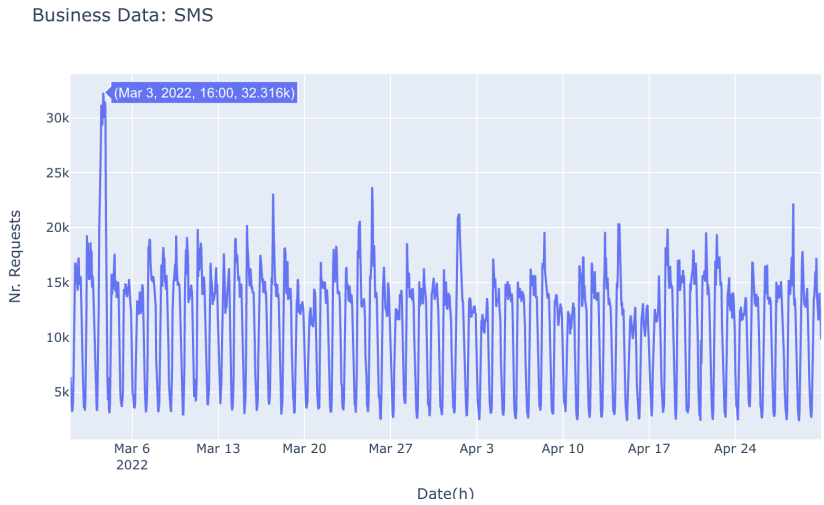


Figure 3.27: Business: SMS

Looking at the Data Sessions in figure 3.28, we can conclude that it is the most used service, reaching more than 8 Million sessions a day. Again, there is no clear pattern here, but unlike SMS and Call, this service had slightly higher values during Easter than the other days. During the day, the lowest peaks are at 4 AM, as per the previously explained metrics, but the highest peaks are at 7 PM or 8 PM, which may be when we leave work. The second lowest peak is at 3 PM, and the second highest is at midnight.

It is now passed on to topup. As for SMS or Voice Calls, it presents a seasonality of a week, but it is visible that there are days in a week with higher or lower topup operations. The lowest values are on Sundays, reaching the highest value around 11 AM and the second highest at 5 PM. The rest of the days during the week presents the highest peak at 7 PM and the second highest at 10 AM. The lowest value is always at 3 AM, and the second lowest is at 3 PM or 4 PM, except for Sundays at 6 PM. So we can also see that during Easter topup operations were

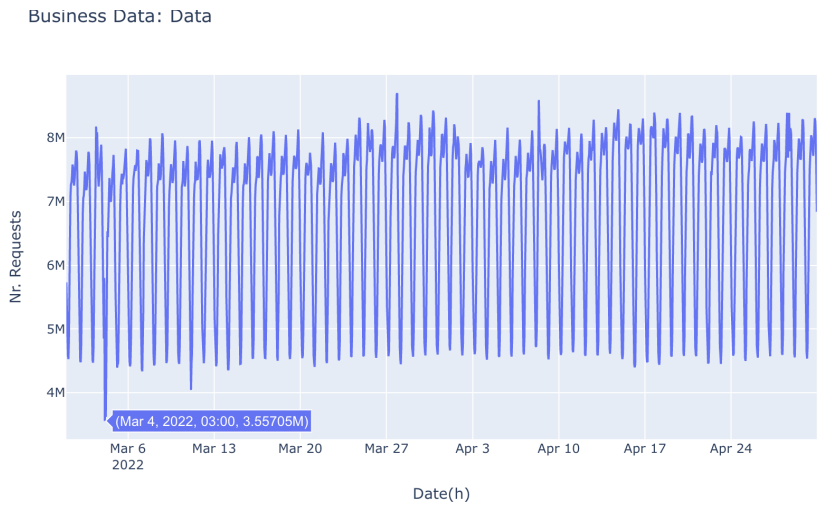


Figure 3.28: Business: Data

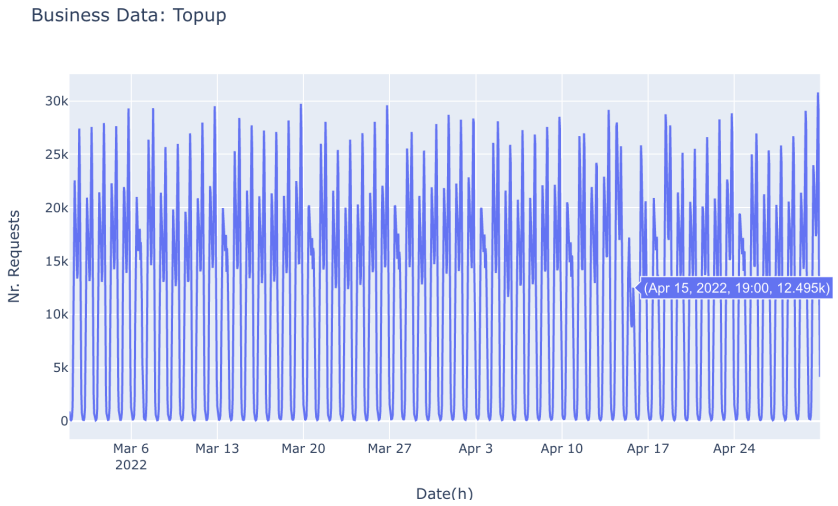


Figure 3.29: Business: Topup

lower.

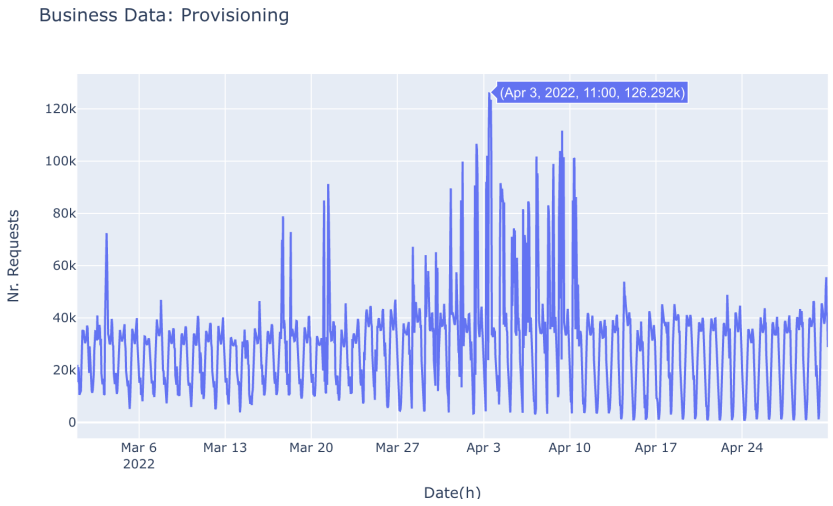


Figure 3.30: Business: Provisioning

Highest, Lowest Values for Business Data				
Metric	Highest	High(2)	Lowest	Low(2)
Call	10AM	3PM	3AM	1PM
SMS	12PM,2PM	7PM,8PM	12 PM,2PM	4PM,5PM
Data Session	7PM,8PM	12PM	4AM	3PM
Topup	7PM,8 PM	12PM	4AM	3PM
Provisioning	Any	4PM,5PM,6PM,7PM	Any	3PM,4PM

Table 3.10: Highest, second highest, lowest, second lowest and seasonality values for Business Data

For the last we have provisioned, here it shows seasonality of 24 hours. The highest provisioning values appear at the end of the month or the beginning of the other. However, more data should be visualized to take that conclusion as a generality. During a week, there is no pattern on which days have higher or lower values. Per day the lowest values are at 4 AM, 5 AM, 6 AM, or 7 AM. As for higher values, they can happen during any hour of the day.

Looking at the Table 3.9, we can see the dissimilarity in quantities of events between the types mentioned above. We can see that Data Sessions have a higher demand compared to the three statistics min, max, and mean. We can relate this fact to our daily routine, the service that we may use with higher frequency is data session, and the lowest one is charging our tariff.

3.5 Summary

This chapter starts by describing how operational and business data is extracted and what approaches will be implemented into the proposed framework. Then, aiming for that goal by the guidance of some use cases Table 3.1 some scenarios when using the tools were exposed, and most of them focused on forecasting.

Following the architecture, requirements are ordered by the MoSCoW approach, where it can be seen from the essential requirements to the ones that are not mandatory within this dissertation goal section 3.2. This analysis helps rank the specificity of the proposed application.

A section with the framework was introduced. We can see in the Figure 3.1 how the process work alongside the forecasting system until the final visual product. This architecture shows user interaction with the forecasting system and the visualization tool using Grafana.

Afterward, the pre-processing and statistics about operational and business data are explained. This section also reported line chart forms of visualizations to give the reader an idea of what type of data we are dealing with. It was shown how each site would pre-process its dataset, what kind of visualizations resulted, and issues were raised. As expected, at night telecommunications have a low rate request and during the day may fluctuate on rush hours, working hours, and weekends. Some tendency-change and peaks that can be evaluated as outliers were distinguished. Here it was concluded that the best feature to forecast is the CPU Load since this feature is the one that shows if machines can handle all the requests every time of day.

It was possible to answer some of the questions proposed on the chapter 1. Relatively to the first questions, it was shown that CPU Load does not evolve in the same way either by looking at a specific site or comparing between sites. Therefore, a forecasting model for each machine is needed since it evolves differently.

Relatively to the second question, there are times of the day when CPU Load has a higher load of requests and when there is no need for so many active resources. As well as when looking at weekends or holidays may also differ.

To answer the third and fourth questions by looking at the correlation between business and operational data can be seen that data session request has more influence on the telecommunication system, reaching more than 8 million requests during a day. A slight difference can be noticed when it comes to its use during the weekends, compared to voice calls and SMS, that the rate request deacreases, on these days the data sessions request increases. As expected when looking at the type of request that leads to error, the data session is the one with higher errors because it is the most used and it is the one with a longer duration type of request

Chapter 4

Forecasting Methodology

Initially, there is the need to understand the difference between prediction and forecasting section 2.3. When we have an input, and our goal is to predict what the result might be giving the input, we are talking about the prediction. However, we are talking about forecasting in this case, where we want to make a future prediction using historical data to estimate future observations.

In order to answer the questions proposed in chapter 1 related to the best forecasting model, this chapter discusses the approaches to applying the best-suited forecasting algorithms studied in the section 2.4 and their variation.

The goal is to develop a pipeline to forecast the values for the next day and, depending on the model, observe which operational features, x, impact the feature chosen y. As a result of the analysis on section 3.4, the feature chosen as y was the average CPU Load. It appears the most inconsistent, with more pattern fluctuation and a higher number of peaks section 3.4. By forecasting the number of waiting for processes or understanding which features correlate with it, we can save resources on the lower demands and be prepared when a higher demand will happen.

This chapter will describe the methodologies applied to make forecasts using machine learning and traditional time series algorithms, how to rank and select the features for the machine learning approach and how to validate the models when presented with a new dataset.

4.1 Proposed System Overview

The process of getting the best parameters per type of forecasting model can be resumed as in the Figure 4.1. However, depending on the model, some adaptations are made before starting the grid search using a selected set of parameters per type of forecasting.

Grid Search is an exhaustive search over a pre-defined set of parameters for an estimator. The search for the best suitable algorithm will be defined by training

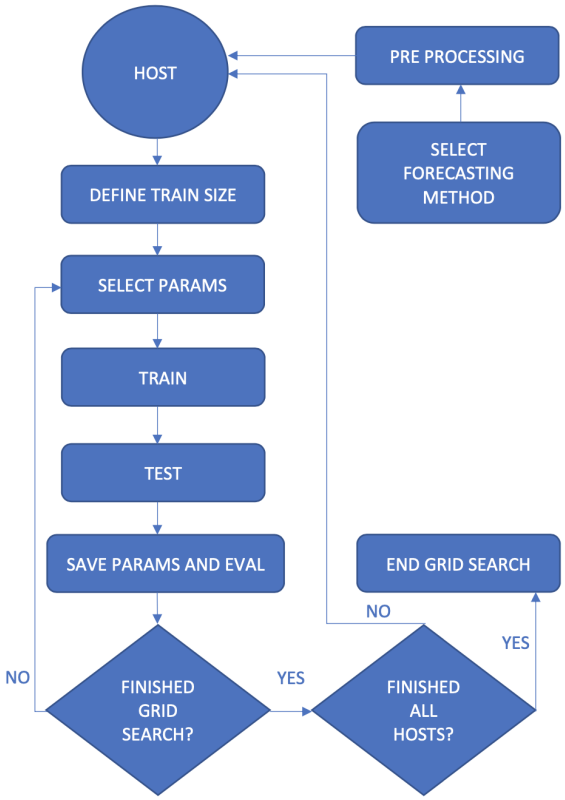


Figure 4.1: Training Grid Search Scheme

depending on the type of training size and tested by forecasting the next 24 hours, as it is the ultimate goal, during one month. The forecasted values are then compared with the testing set, the actual values. The grid search will be performed over all machines for all sites, testing the pre-defined set of parameters.

As we can see in the Figure 4.1 for each host or machine, we are going to define the type of training size as will be seen in the following sections, where we can have a fixed training size or an expanding one. Next, for each iteration, inside the grid search function, for each set of parameters, we will train, test, evaluate, and save the results until all parameter combinations are finished for all machines. Ultimately, we store the best set of parameters and its Mean Absolute Percentage Error (MAPE).

Here are tested simple and complex models to see how it behaves throughout the hosts. The aim is to classify the best model by the time it takes to train, with the lowest MAPE for overall and outliers error.

4.2 Feature Importance and Selection

A dataset with multiple features can benefit the forecasting result section 2.4. However, at the same time can lead to worse results. In order to avoid that, a feature selection can improve the forecasting results in different ways, such as reducing training time, simplifying complexity, improving the model's accuracy, and even avoiding over-fitting by eliminating unnecessary features [14].In subsection 2.4.1 indicates different types of feature selection, such as filter methods, wrapper methods, and embedded methods.

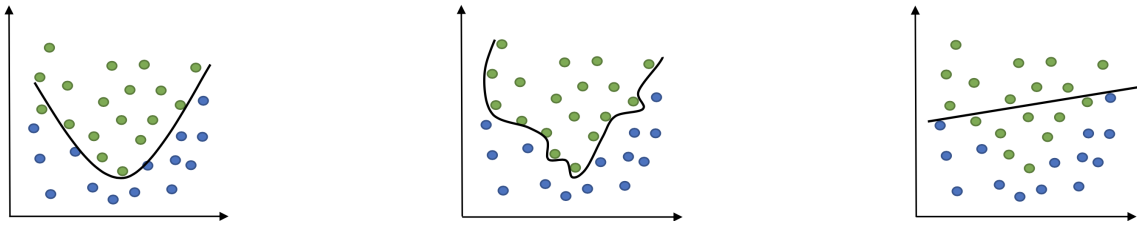
We can look at this problem as an optimization problem. It needs to define a cost function to be minimized subject to certain conditions and an optimization strategy to explore the search space. We may encounter two problems when selecting the features: relevance and redundancy [14]. When ranking the features by importance, we encounter the relevance problem based on their classification performance. Nevertheless, the way we select the set of features may lead to redundancy. For example, a feature may appear in a higher rank of importance, but a set of features may turn out to be an irrelevant feature. The opposite case may appear where a weakly relevant feature in a set of features may be relevant.

For our approach, it was used an embedded method to rank features. For a better explanation, it can be found on subsection 2.4.1. As explained in the embedded method, the algorithm has its feature selection logic, so this process is performed while training the model. The algorithm used was Random Forest, resulting in a ranked list of features.

After the features ranking process, multiple strategies select a lower number of features to decrease the complexity or avoid overfitting the result by using all of the variables as explained in section 2.3. In this dissertation, it was opted to use an elbow graph to contribute to the feature selection. Then, an iteration through all the features is done. In each step, a prediction algorithm is trained with the accumulative features to save the selected error metric at each iteration. The outcome is a graph that can look like an elbow graph since, in the beginning, the more features we add, the lower the error gets, and after accumulating a certain number of features, it starts increasing. With this strategy's support, we can see which features contribute positively by having an acceptable error.

4.3 Train and Test

There are two types of model validation which are out-of-sample and in-sample performance. The out-of-sample validation, called generalization performance as well, on unseen data, measures how well a trained learning algorithm generalizes from the sample set it was trained on [31]. The in-sample validation refers to an empirical error or training error. Generalization is an essential concept in mind since the subsequent model to be applied in a real scenario may encounter variations in patterns and outliers.



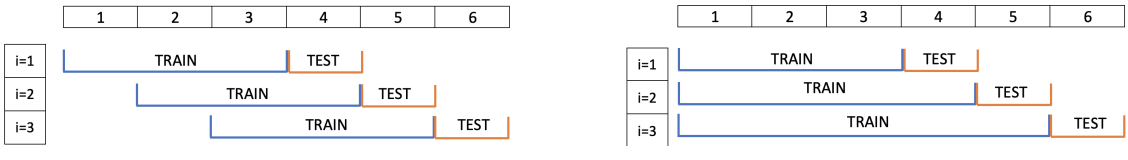
(a) (b) (c)

Figure 4.2: Bias-Variance Trade-off Example: (a) feature space with optimal segregation (b) overly complex model (c) overly simplistic model

For example, in a scenario like Figure 4.2 where machine learning classification linearly separates two classes depending on the model and the data it is tested with, we may come across some challenges. Looking at Figure 4.2c, we see a linear separation classification between two classes. Since the problem is not always obvious, it can be underfitting data. This error is called bias, and it arises when the model pays very little attention to the training data, it leads to a high error in the training and test data. Bias may occur when the model is too simple and trained with few features.

When we look at Figure 4.2b, we have an example of high variance, pay much attention to training data, and do not generalizes on the data which it has not been seen before. As a result, it performs very well on training data but results in a high error for unseen data. In order to reduce this error, more training data should be introduced.

So, as the bias increases, variance decreases, and vice-versa, this is called a trade-off. The goal is to get a model with low bias and variance.



(a) (b)

Figure 4.3: Window Length Approaches (a) sliding window (b) expanding window

Since the model may be used in a real-life scenario and the dataset increases daily, fixing a taining size may be helpful on the long road. Two techniques are used for the machine learning training: expanding and sliding window, as seen in the Figure 4.3.

On the Figure 4.3a, we can see that the training size of the first, second, and third iteration (i=1,2,3) is still the same: three units. On the other side, when looking at figure Figure 4.3b, we can see the training size increasing each iteration as new data arrives. In this case, all data is essential to forecast or predict the next step.

The choice between the training size reflects the seasonality of the data. When a smaller training size is used, the patterns are shorter than an expanding training size. The fixed training size allows new patterns to be developed with time. When an expandible size is used, the model has a change to look for longer-duration patterns, for example, at yearly holidays instead of monthly or daily patterns. This approach can be beneficial for stationary data to make maximum use of available training data at each point in time [31].

4.4 Forecasting

In the context of time series data, one variable is the value at time t, and the other is any value at time t-x, where x is a non-zero positive integer [14]. Meaning that the forecast depends on the historical data. Given a vector of past observations V = v(t-1),...,v(t-N) of metrics for example CPU Load the forecasting problem consists of predicting values at time t denoted by v(t). "The Short-Term forecasting or prediction focuses on a time frame or period of fewer than three months, whereas the Mid-Term period focuses on a time frame of three months to one year and the Long-Term considers a time period more than a year."Mahalakshmi et al. [1]

We can apply two types of forecasting: machine learning or traditional time series models. There is some difference between these models depending on the input and output. In the case of time series forecasting, they can fit into two groups, univariate and multivariate. When a time series is composed only of a variable varying in time, it is called univariate. When the time series is composed of more than one variable also varying in time, it is called multivariate. As we are in the context of forecasting, we need to pre-process data differently according to the forecasting models.

The pre-processing to prepare data for time series approaches is pretty much deciding on the training and forecasting sizes. These models can use tendency, seasonality, or different levels of importance to apply these techniques successfully. There is difficulty in applying the model type, and higher expertise may be needed to handle its parameters.

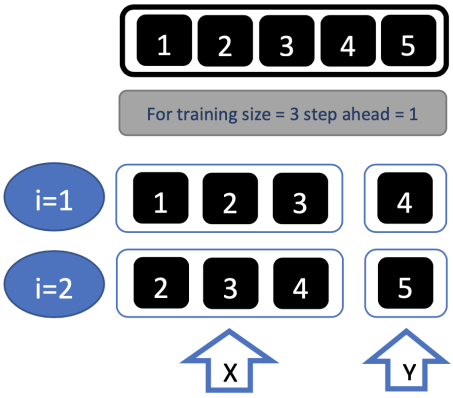


Figure 4.4: Univariate Preprocessing Example

When it comes to using machine learning, in our case regressor, it is not that obvious as these models are prepared to estimate a particular event based on a set of features, not necessarily concerned about the future section 2.3. In the contemplation of forecasting, there is the need to transform time series into supervised learning, meaning that the machine learns from the labeled data. We can see in Figure 4.4 and Figure 4.5 examples of how can to prepare univariate or multivariate data for forecasting using machine learning.

In the Figure 4.4, we have an example of how we can prepare univariate time series for supervised machine learning. In this case, we have a time series composed by [1,2,3,4,5], independently of time but keep in mind that they are sequential. If the goal is to train a machine learning algorithm with size three and to forecast one value into the future, then we start by taking the first three values as the first row of the X, our input, and the fourth value is the Y, the output. The second raw of the X will be formed starting with the second value until the fourth because of the training size, and Y is the fifth value. This way, the training dataset is constructed. Independently on the time unit, either 3 hours or 3 minutes, the fourth value results in the output value. In the case of using this type of approach, we are eliminating all the seasonality, trends, or time variables, and it represents phenomena observed at a single point in time.

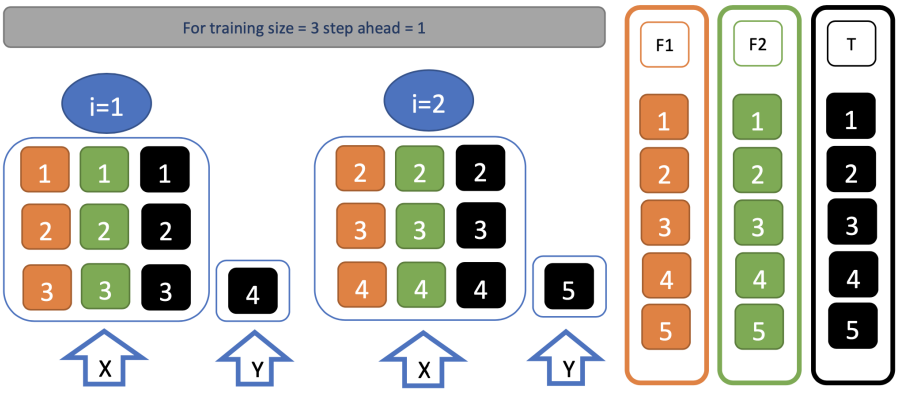


Figure 4.5: Multivariate Preprocessing Example

In the case of multivariate, as shown in Figure 4.5 can be a little bit more chal-

lenging to understand, but it follows the same rationale. The goal here is to train a machine learning model with size two and forecast 1 step ahead, using a multivariate time series. Data here is composed of three variables, F1 and F2 will be used to train data in the direction of forecasting the variable T for the next day. The difference is that three-time units are merged data of both F1 and F2 with the three values from T, creating the X using the three features to predict the T value.

Aside from the data adaptation from univariate and multivariate data, there are two types of outputs, Y. They can be a single output or multi-out, meaning that for single output, as can be interpreted from the name, only one value is forecasted. For multi-out, we are forecasting multiple values each time. In the case of traditional time series techniques, we only need to explicitly express a number of forecasting values we want to receive from the models. However, some machine learning regressors are not prepared for multiple outputs, and there are some alternatives to deal with this dilemma. When it comes to supervised algorithms that can forecast multiple outputs, for example, Long Short-Term Memory (LSTM), as in the section 2.4, was used as a forecasting algorithm. There is a slight difference in the pre-process shown before. Instead of training with Y of only one value, if, for example, Figure 4.5 or Figure 4.4, with a step ahead=2 then for i=1, the test set it Y=[4,5]. When the algorithms cannot output multiple values simultaneously, we use the sliding window approach explained in section 4.3 on Figure 4.3a, from where we use the output from previous forecasting values until reaching the number step ahead of desired. For example, in the case where step ahead=2 and training size=3 to forecast the last value, we use two real values and the last forecasted value.

One big difference between forecasting using machine learning or a traditional time series algorithm is how the train takes effect in the future. Although for machine learning, we save the trained model and re-train with new values, when it comes to time series, there is no need to save the model for future forecasting; nevertheless, we need to re-train with all historical values and the new ones. Another big difference is that using traditional time series techniques, we only use one variable, which is the one to forecast. On the other hand, the main advantage of using machine learning is the ability to use multiple variables or features to help with forecasting.

4.5 Evaluation Metric

The evaluation metrics that will be used are Mean Square Error (MSE) because we want to penalize the peaks and MAPE for assisting MSE for a better interpretation. In evaluating the algorithm, we can compare it in several ways. One is the overall error, the total error by a time section, or extracting the error only at outliers. This last error calculation is helpful because it gets lost on the mean when we do all over the time series since there are fewer outliers than within the range calculated to get the outlier. The error calculated on the values without outliers helps interpret how well a model performs for an ideal case, where all values stay under the limits.

In section 2.3 was elucidated on how to characterize the outliers based on the mean and standard deviation using k=[1.5, 1.75 ,2] for varying degrees of stringency.

During this dissertation, CPU Load will be forecasted since it is the feature with the most inconsistency and more outliers. This is a feature for forecast since, in a case where values are too high, meaning that it could be a bottleneck within the telecom system. On the other hand, if values are too low, the resources are not being utilized, meaning they could be decreased.

Outliers Count per Machine (CPN)			
Machine	k=2	k=1.75	k=1.5
1	51 ≈ 1.47%	150 ≈ 4%	424 ≈ 12%
2	33 ≈ 0.95%	107 ≈ 3%	374 ≈ 11%
3	30 ≈ 0.86%	111 ≈ 3%	404 ≈ 12%
4	31 ≈ 0.89%	116 ≈ 3%	383 ≈ 11%
5	25 ≈ 0.72%	98 ≈ 3%	374 ≈ 11%
6	18 ≈ 0.52%	110 ≈ 3%	378 ≈ 11%
7	20 ≈ 0.57%	106 ≈ 3%	399 ≈ 11%
8	21 ≈ 0.6%	99 ≈ 3%	388 ≈ 11%
9	45 ≈ 1.29%	122 ≈ 4%	399 ≈ 11%
10	46 ≈ 1.32%	131 ≈ 4%	396 ≈ 11%
11	35 ≈ 1.01%	126 ≈ 4%	415 ≈ 12%
12	40 ≈ 1.15%	123 ≈ 4%	414 ≈ 12%
13	40 ≈ 1.15%	145 ≈ 4%	412 ≈ 12%
14	43 ≈ 1.24%	143 ≈ 4%	437 ≈ 13%
15	38 ≈ 1.09%	111 ≈ 3%	381 ≈ 11%
16	35 ≈ 1.01%	124 ≈ 4%	414 ≈ 12%

Table 4.1: Outliers Count per Machine CPN: higher values colored in red and lower values colored in green each threshold

We can see outliers in the tables 4.1, 4.2 and 4.3 where the outliers are counted per machine and therefore per site. We can see that we have fewer values for higher values of k, and for lower values of k results in more outliers. Considering that looking at the operational data size in Table 2.2, for a total of 6 months, data of

Outliers Count per Machine (DSGW)			
Machine	k=2	k=1.75	k=1.5
1	10 ≈ 0.29%	23 ≈ 1%	173 ≈ 5%
2	9 ≈ 0.26%	24 ≈ 1%	154 ≈ 4%
3	7 ≈ 0.2%	18 ≈ 1%	187 ≈ 5%
4	9 ≈ 0.26%	17 ≈ 0%	188 ≈ 5%

Table 4.2: Outliers Count per Machine DSGW:higher values colored in red and lower values colored in green each threshold

Outliers Count per Machine (SMP)			
Machine	k=2	k=1.75	k=1.5
1	66 ≈ 1.9%	266 ≈ 8%	496 ≈ 14%
2	57 ≈ 1.64%	262 ≈ 8%	506 ≈ 15%
3	72 ≈ 2.07%	289 ≈ 8%	489 ≈ 14%
4	105 ≈ 3.02%	210 ≈ 6%	392 ≈ 11%
5	88 ≈ 2.53%	178 ≈ 5%	375 ≈ 11%
6	108 ≈ 3.1%	213 ≈ 6%	416 ≈ 12%

Table 4.3: Outliers Count per Machine SMP: higher values colored in red and lower values colored in green each threshold

around 69 120 records for CPN, 17 280 for DSGW, and 25 920 for SMP, we do not have a high number of outliers. For CPN with around 440 outliers corresponds to 0.64% of data, DSGW with around 200 outliers corresponds to 1.15%, and SMP with around 1.8%.

4.6 Summary

This chapter introduced the methodology employed for this dissertation. We saw that we have two different techniques with the aim of forecasting. We can do it by using machine learning with some additional data pre-processing or traditional time series techniques. We start by presenting the system overview in the Figure 4.1 and then explaining how each operation will proceed. First, an exhaustive grid-search will be performed per machine and training size to obtain the parameters that lead to the model with lower MSE, per model. The tests were done using a grid-search approach by training each model with three months corresponding to October, November, and December and one month to test, corresponding to January. In the end, the model with the lowest error for each type of forecasting algorithm is saved for each machine at all sites. Subsequently, using all the best models forecasts the new data from February until April and uploads them to the final dashboard in Grafana.

Chapter 5

Forecasting Results

In the chapter before, the methodologies and the pipeline are described. In this chapter, it is going to be shown the results in-depth manner. First, it is going to be shown how the process of choosing the right features by their importance, forecasting using Long Short-Term Memory (LSTM) for both univariate and multivariate data, and forecasting using Seasonal Autoregressive Integrated Moving Average (SARIMA) and Triple Exponential Smoothing or Holt-Winters (TES) for both expanding and sliding window. The last questions proposed in the chapter 1 are answered in this chapter. Finally, the best model for each forecasting algorithm is explained and presented with the final dashboard and its properties.

5.1 Feature Importance and Selection

Feature importance is a technique to assign a score for each feature used as input based on how useful they are at predicting a target variable. Features are ranked by the correlation with the target feature. This is an important step since next, the set of features is used to predict the target, which is the avg_cpu_load. Feature selection is made because the number of features per site is high, 108 for DSGW, 104 for CPN, and 64 for SMP, as seen on Table 2.2. Using all of them can increase the model's complexity, training time, and overfitting. Random Forest (RF) is used to rank features, followed by a features selection using RF to predict and employ the elbow graph and get the selected features.

The process of ranking feature is by training, in this case, RF and testing it. First, the features will be internally ranked by their importance in the prediction compared to the test set. After getting the set of all features ranked by their importance, the next step is selecting the features using an elbow graph.

Elbow Graph is produced by iterating over the features ranked chosen. Each iteration accumulates one feature at a time to train and test using RF. In the end, we choose the set of features with lower error. In the Figure 5.1 we can see an example the resulting graph.

In the Figure 5.1, we can see that the more features we add, the lower the error

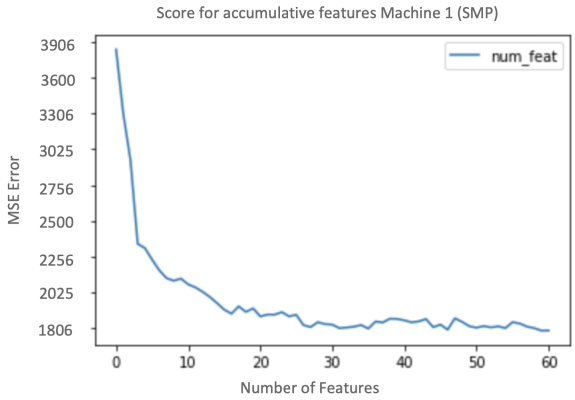


Figure 5.1: Elbow Graph for Machine 1 (SMP)

gets until a certain point. We can also see that the difference between adding one more feature at the begging is higher than at the end. Also, in this figure, the lowest error is when we use 59 features, but the error using 28 and 35 is almost the same. At this point, we need to decide if this small error is worth it when forecasting since it adds more complex the model and, therefore, the longer it takes to perform the forecast. In the case of this dissertation, it will be considered the absolute minimum value, even if it takes longer in the interest of reaching the best estimation of the actual values when forecasting.

After this process, let us look at the most important features of each site. To do this it was counted how many times a feature was selected for each machine for CPN Figure 5.3, for DSGW Figure 5.2 and for Figure 5.4.

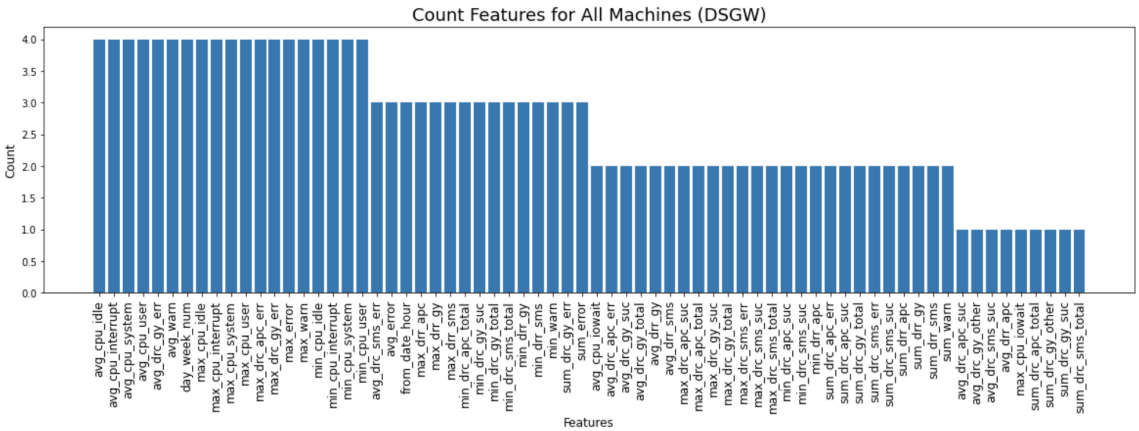


Figure 5.2: Count Feature for All Machine (DSGW)

In figure Figure 5.2, the features selected for DSGW are presented, and it can be seen that 20 features are grouped into the most important features for all the machines. Those features are: "avg_cpu_idle", "avg_cpu_interrupt", "avg_cpu_system", "avg_cpu_user", "avg_drc_gy_err", "avg_warn", "day_week_num", "max_cpu_idle", "max_cpu_interrupt", "max_cpu_system", "max_cpu_user", "max_drc_apc_err", "max_drc_gy_err", "max_error", "max_warn", "min_cpu_idle", "min_cpu_interrupt", "min_cpu_system" and "min_cpu_user".

Figure Figure 5.2 presents the features selected for CPN. Three features are grouped into the most important features for all machines. Those features are: "avg_cpu_idle",

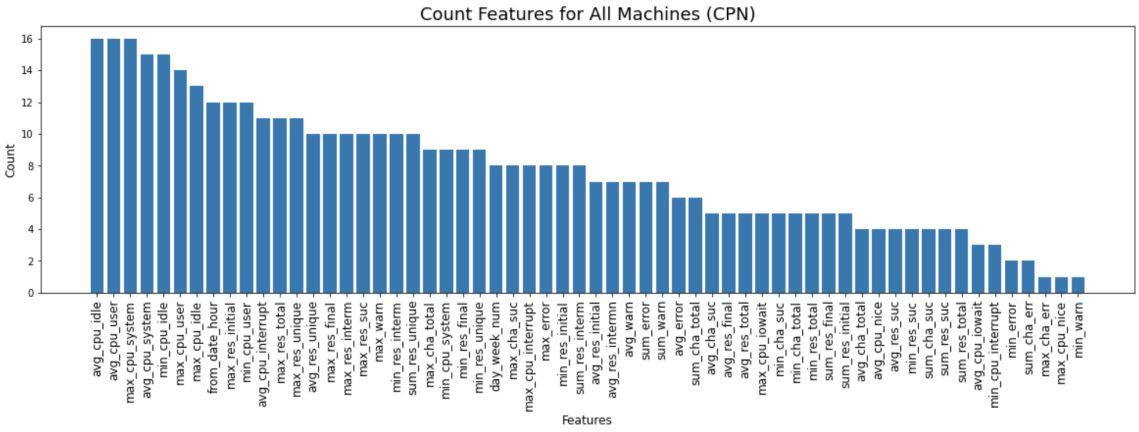


Figure 5.3: Count Feature for All Machine (CPN)

"avg_cpu_user" and "max_cpu_system".

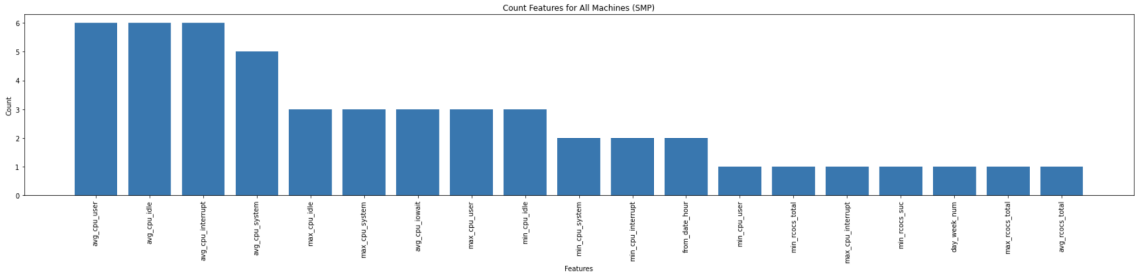


Figure 5.4: Count Feature for All Machine (SMP)

Figure Figure 5.4 presents the features selected for SMP. Three features are grouped into the most important features. Those features are: "avg_cpu_idle", "avg_cpu_interrupt" and "avg_cpu_user".

These graphs show that the essential features are most of the CPU statistics and errors created from data sessions and voice call evens. The pre-process was exhibited in section 3.4. In this sub-section, data sessions were the type of request with the highest number of errors, which may contribute to the selection of the selected features. Moreover, another feature, which was later added, is the day of the week. It goes according to the conclusions in section 3.4 since on Saturdays or Sundays, fewer requests are made.

5.2 Forecasting using ML

For the machine learning forecasting LSTM was the best suited since it is the model with the best performance, from the Table 2.5

LSTM is a Recurrent Neural Network (RNN) that can learn and forecast long sequences in detail can be seen at section 2.3. The training set is composed of the first three months, and the test set by one month from the dataset. The evaluation metric is Mean Square Error (MSE) because we want to penalize peaks however the metric showed is Mean Absolute Percentage Error (MAPE) because of a better interpretation.

There are different pre-processes for each type of dataset, depending on the number of features. Let us look at the univariate data when forecasting data using an expanding window with a continuous update for all data.

5.2.1 LSTM - Univariate Results

The parameters used for this model can be seen on Figure 5.5 which are very similar as on [14].

Layer (type)	Output Shape	Param #
lstm_3 (LSTM)	(None, 48, 50)	10400
lstm_4 (LSTM)	(None, 48, 50)	20200
lstm_5 (LSTM)	(None, 50)	20200
dense_1 (Dense)	(None, 24)	1224

Figure 5.5: LSTM Structure of Univariate with MultiOut: 3 layers, each with 50 neurons, and a final dense layer of 24 values.

These parameters are used for univariate and multivariate comparisons, changing only the network's ability to receive a different input, whether with other features' impact or not.

To validate the LSTM model using univariate data, the initial training and test set is used, from which one percent of the test set is used for internal validation purposes.

We can see that the training and validation error follows the same pattern and that the error decreases fast until almost zero MSE, which means that it is a good model for our data. Looking at the forecasting result for new data in Figure 5.7 can be seen that the forecast maintains the same patterns. Although it maintains the same pattern, because there are few outliers, the error is not that high as it results in 13.4% MAPE and 6735.73 MSE.

In figure Figure 5.8 we can see how applying the structure on Figure 5.5 for all dataset updating the model each 24h. We can see the error for this machine in the table Table 5.1 that it increases to 16.98% for the full dataset because it also presents more outliers. Analyzing the outlier points for the different kinds of

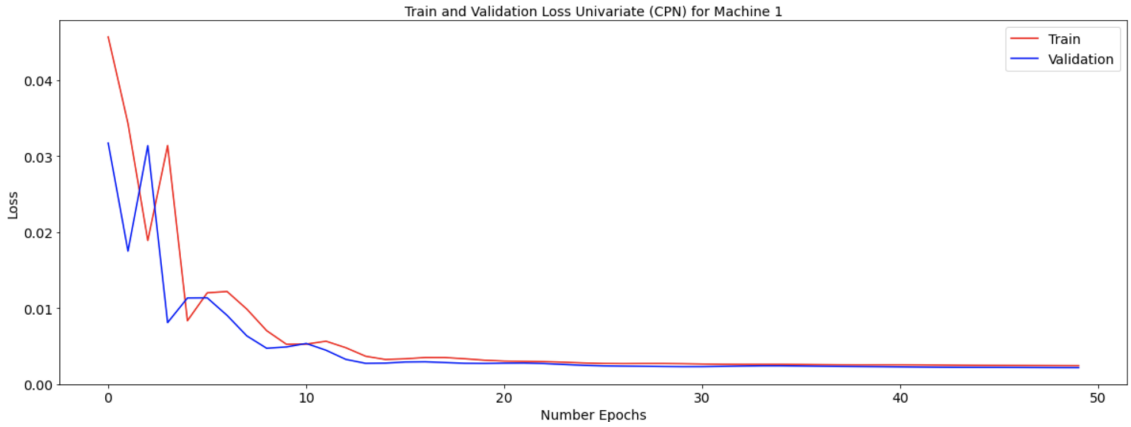


Figure 5.6: Train and Validation Loss for Univariate Data (CPN): Loss corresponds to MSE error

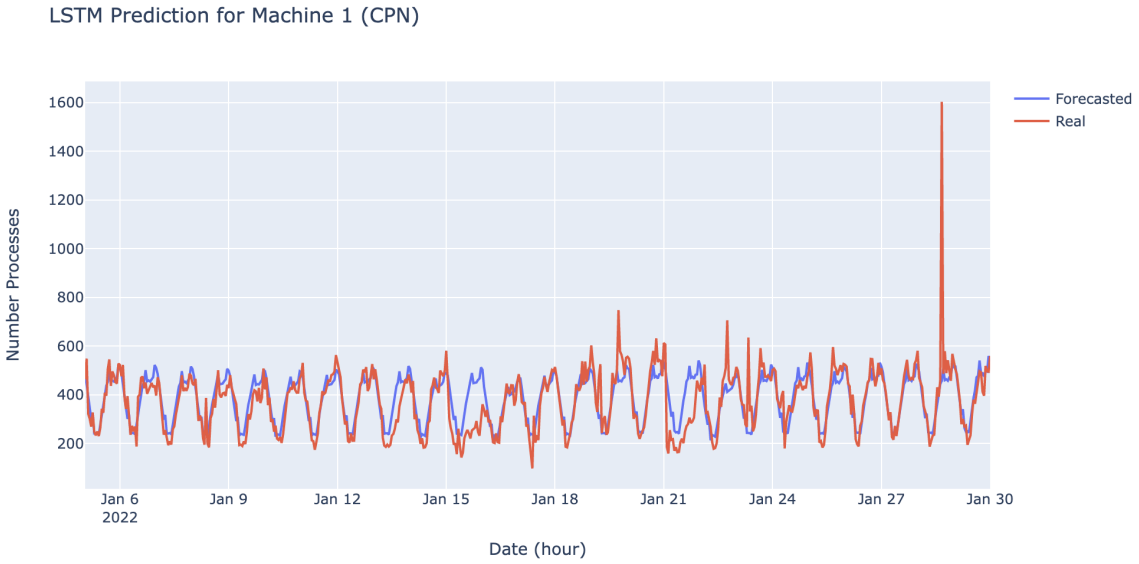


Figure 5.7: Forecast and Real values of univariate test dataset for CPN Machine 1 - TVT

thresholds, we get MAPE is 29.22% for the lowest threshold, for the medium is 31.96% and for the highest is 46.36%. The error increases with the threshold because fewer points are analyzed than those with the highest error. We can also verify if the model has good performance regarding its periods and seasonality when obtaining the error without the anomaly points, meaning how the model behaves in a perfect case. For example, in the Table 5.1, we can see that the error is a lot lower, around the overall error, which was expected, with 16.31%. Removing the outliers above the lowest threshold, 17.21% above the medium, and 17.44% above the highest threshold.

5.2.2 LSTM - Multivariate Results

We also can use more features to add information and eventually decrease errors. Let us see if the performance increases when adding more features using

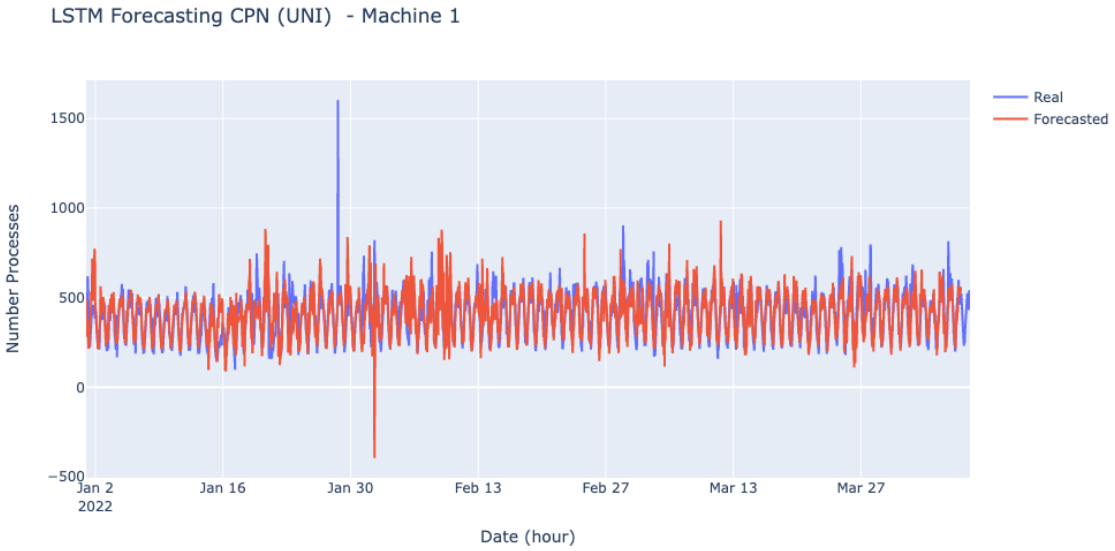


Figure 5.8: LSTM Forecasting for Machine 1 CPN - Univariate

the multivariate data.

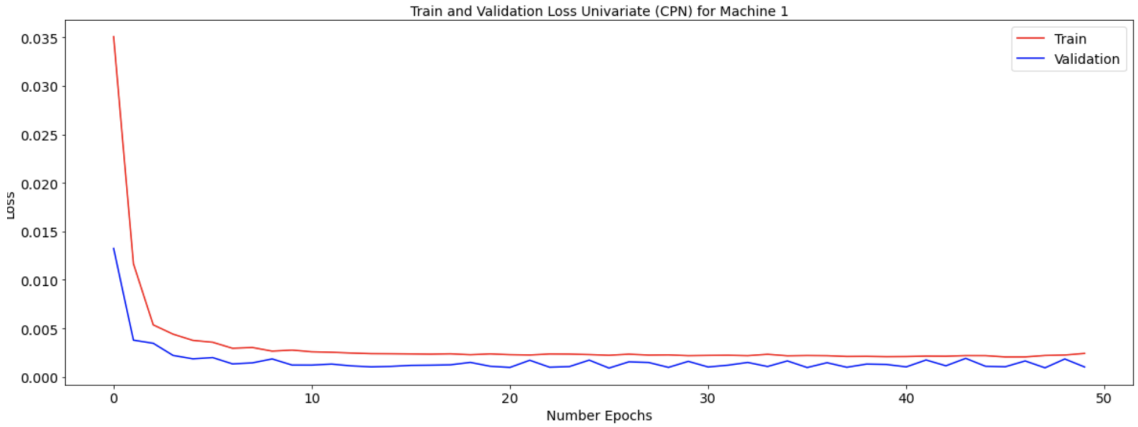


Figure 5.9: Train and Validation Loss for Multivariate Data (CPN): Loss corresponds to MSE error

In the Figure 5.9 we can see how the MSE of the training set and validation set behaves. It is more regular than the one in the previous sub-section since this dataset's learning rate was low enough because it almost reached zero and not too high, making the error vary.

In the Figure 5.11 we have a forecasting using LSTM which resulted with the following errors MSE=4411.9 corresponds to MAPE=11.4%. Compared to the previous result, it got better; on the validation part, it resulted in 13.4%. However, let us see how it behaves during all data.

The overall error of machine one from site CPN in Table 5.1 decreased compared to the forecasting with univariate data to 15.4%. The error at the outliers site for almost all the thresholds got lower with a MAPE of 24.8% for the lowest thresh-

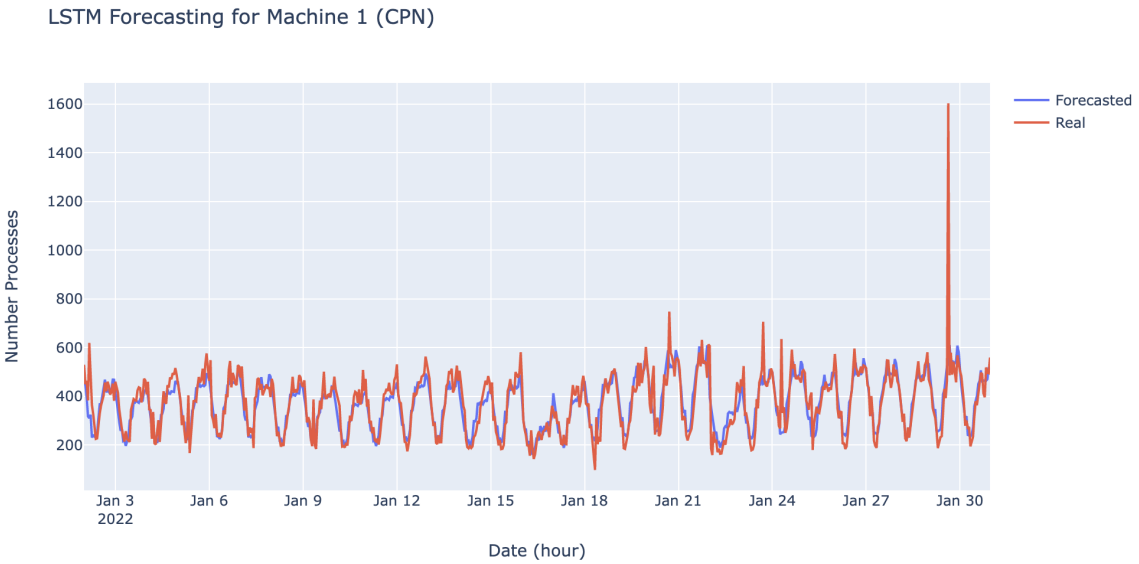


Figure 5.10: Forecasting and Real Values of multivariate test dataset for CPN Machine 1 - TVT

old, 32.73% for the medium, and 43.66% for the highest. Comparing the forecasting without outliers, it gets around the overall error and is still lower than univariate data, with 15.13% removing the outliers above the lowest threshold, 15.54% above the medium threshold, and 15.89% for the highest.

Still in the Table 5.1 for CPN, in the Table 5.2 for DSGW and in the Table 5.3 we proved that using the features selected for each machine at each site improved the forecasting results for almost all the cases. The error was almost the same for the cases that did not improve.

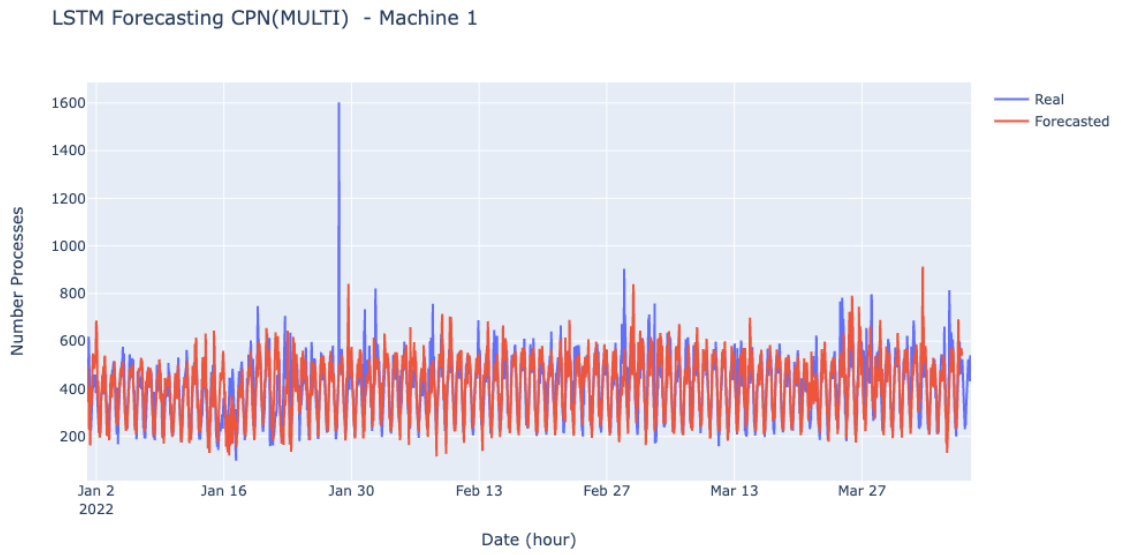


Figure 5.11: LSTM Forecasting for Machine 1 CPN - Multivariate

LSTM Forecasting MAPE (OvE), (OtE), (NOE) CPN								
Mch.	Type D.	OE	Outliers Error			Non Outliers Error		
			k=1.5	k=1.75	k=2	k=1.5	k=1.75	k=2
1	UNI	16.98	29.22	31.96	46.36	16.31	17.21	17.44
	MULTI	15.4	24.8	32.73	43.66	15.13	15.54	15.89
2	UNI	9.44	13.78	21.72	26.96	9.97	10.02	10.23
	MULTI	9.13	13.72	21.33	27.52	9.63	9.72	9.92
3	UNI	11.24	15.43	23.5	24.84	11.75	11.84	12.07
	MULTI	11.4	14.9	22.12	29.66	12.0	12.04	12.2
4	UNI	11.89	15.81	20.86	24.31	12.44	12.52	12.71
	MULTI	10.81	15.94	19.88	22.51	11.23	11.45	11.64
5	UNI	10.58	16.15	20.62	34.35	10.99	11.24	11.3
	MULTI	10.41	16.33	23.13	36.3	10.79	11.0	11.12
6	UNI	11.3	14.69	19.39	43.11	11.92	12.0	12.11
	MULTI	10.32	13.97	19.22	31.49	10.93	11.01	11.18
7	UNI	9.95	12.24	16.67	24.33	10.7	10.73	10.83
	MULTI	9.56	13.04	18.55	32.04	10.16	10.28	10.41
8	UNI	10.89	13.27	18.04	29.1	11.63	11.61	11.7
	MULTI	10.3	13.61	18.85	32.56	10.93	10.99	11.09
9	UNI	12.04	16.86	27.0	39.9	12.45	12.42	12.55
	MULTI	13.21	18.63	29.33	42.4	13.53	13.53	13.7
10	UNI	12.41	17.23	22.44	32.08	12.83	12.98	13.02
	MULTI	12.45	20.06	30.66	44.58	12.53	12.72	12.86
11	UNI	9.6	12.47	16.74	29.19	10.28	10.3	10.33
	MULTI	11.87	15.73	21.27	36.88	12.38	12.45	12.51
12	UNI	13.0	16.38	23.47	32.17	13.55	13.53	13.63
	MULTI	11.64	15.94	25.7	36.6	12.07	12.04	12.2
13	UNI	10.75	14.83	21.26	34.14	11.25	11.25	11.38
	MULTI	10.85	16.33	22.85	34.7	11.18	11.29	11.48
14	UNI	13.34	20.15	28.03	57.34	13.43	13.69	13.81
	MULTI	12.58	17.51	24.38	51.39	12.93	13.05	13.11
15	UNI	10.99	13.44	17.3	20.77	11.72	11.72	11.81
	MULTI	11.25	13.53	18.53	22.61	11.99	11.94	12.05
16	UNI	12.32	16.17	22.24	27.43	12.84	12.89	13.08
	MULTI	11.15	14.86	22.54	33.48	11.7	11.68	11.86

Table 5.1: LSTM Forecasting CPN between January and April (OvE) overall Error, (OtE) ouliers error for each threshold and (NOE) non outliers error for the lowest threshold: for each type of data (Type D.) which can be univariate (UNI) or multivariate (MULTI). For each machine, the lowest MAPE between the univariate and multivariate MAPE is colored in green

LSTM Forecasting MAPE (OvE), (OtE), (NOE) DSGW								
Mch.	Type D.	OE	Outliers Error			Non Outliers Error		
			k=1.5	k=1.75	k=2	k=1.5	k=1.75	k=2
1	UNI	10.94	23.15	54.03	45.09	11.21	11.51	11.74
	MULTI	11.19	23.19	48.83	50.39	11.47	11.81	11.98
2	UNI	8.75	16.37	35.99	53.62	9.38	9.54	9.56
	MULTI	9.9	20.51	41.02	54.55	10.37	10.65	10.7
3	UNI	11.23	17.57	53.12	24.85	11.81	11.85	12.12
	MULTI	9.32	16.81	45.8	23.36	9.84	9.99	10.22
4	UNI	9.9	17.56	55.23	52.33	10.39	10.56	10.69
	MULTI	9.02	16.31	52.37	48.54	9.54	9.7	9.82

Table 5.2: LSTM Forecasting DSGW between January and April (OvE) overall Error, (OE) outliers error for each threshold and (NOE) non outliers error for the lowest threshold: for each type of data (Type D.) which can be univariate (UNI) or multivariate (MULTI). For each machine, the lowest MAPE between the univariate and multivariate MAPE is colored in green

LSTM Forecasting MAPE (OvE), (OE), (NOE) SMP								
Mch.	Type D.	OvE	Outliers Error			Non Outliers Error		
			k=1.5	k=1.75	k=2	k=1.5	k=1.75	k=2
1	UNI	8.98	12.04	12.47	19.35	9.57	9.7	9.72
	MULTI	8.33	11.67	12.25	20.56	8.89	9.03	9.04
2	UNI	9.52	13.38	13.32	25.52	9.96	10.23	10.2
	MULTI	8.41	11.63	12.11	22.87	8.97	9.14	9.13
3	UNI	9.28	13.16	12.92	21.77	9.73	9.98	9.94
	MULTI	8.7	13.1	12.93	22.2	9.08	9.36	9.35
4	UNI	12.43	26.32	32.58	44.28	11.63	12.05	12.28
	MULTI	10.92	25.1	30.44	42.23	10.11	10.6	10.81
5	UNI	10.37	22.37	31.47	44.42	9.91	10.17	10.37
	MULTI	10.77	23.82	34.26	45.72	10.18	10.44	10.75
6	UNI	12.43	24.43	31.51	42.44	11.89	12.21	12.32
	MULTI	10.99	24.04	31.74	40.65	10.33	10.68	10.92

Table 5.3: LSTM Forecasting SMP between January and April (OvE) overall Error, (OtE) outliers error for each threshold and (NOE) non outliers error for the lowest threshold: for each type of data (Type D.) which can be univariate (UNI) or multivariate (MULTI). For each machine, the lowest MAPE between the univariate and multivariate MAPE is colored in green

5.3 Forecasting using traditional TS Algorithms

For traditional time series forecasting SARIMA and TES were the best suited algorithms. They were the ones with lower errors and the ones that approximate the estimated values to the real ones section 2.4. The main difference between TES and SARIMA is the data weights section 2.3, it allows a exponential decrease according to how recent they are.

These techniques are used for univariate data, meaning one variable, and there is no need for additional pre-processing in order to use them. SARIMA and TES were tested in an expanding and sliding window scenario. It is assumed that when using an expandable window, we get a better approximation of the forecasted values to the real ones because we are allowed to learn from more extended patterns. For the sliding window approach, a grid-search was done to get the fixed training size that leads to lower MSE.

5.3.1 SARIMA Results

The first step to analyzing time series is identifying its components, which can be used, Box Jenkins, in section 2.3, or Grid-Search, which is usually the most used. Like it was done for LSTM on the subsection before section 5.2, we will use the final model definition based on the values until the 30th of January, which corresponds to 53% of data.

Box Jenkins (BJ) was used to demonstrate and cut the length of the Grid-Search by not testing all the parameters. Here the seasonal period in samples (S), the seasonal differencing order (D), the tendency differencing order (d), the orders (P and Q) of the seasonal part, and the orders (p and q) of the tendency part are defined. Parameters are defined in order to turn Time Series (TS) stationary and use the forecasting time series models. Usually, no more than d=1 and D=1 is necessary, but it is possible to apply higher differencing orders.

- First step is verifying if the TS is stationary. If so, d=0 and D=0, we do not need to apply the differencing operation.
- Second step, if the series is not stationary and has a trend applying simple differencing operation, d=1. If the trend is not removed, then apply d=2 and test again.
- Third step, with the trend removed from the second step, if seasonality is present, the Autocorrelation (ACS) should present periodic peaks at each S lag. If no peaks appear, then seasonality does not exist, and D=0. Otherwise, apply a first-order differencing, D=1. If ACS peaks disappear, then D=1. Repeat this step if there are still peaks.
- Fourth step, we have to define p,q, P, and Q. The ACS and Partial-autocorrelation (PACS) are analyzed to determine the appropriate orders. For p and q, we

should observe ACS and PACS between t=1 and T<S and observe the values which fall out of the confidence bounds. The criteria for choosing these parameters should be the Principle of Parsimony: "Adopt the simplest acceptable model ." For P and Q, we can resample the ACS and PACS at each S lag and perform the same logic as before.

Autocorrelation is the correlation between two observations at different instances in a time series. When there is correlation means that past values influence the present or future. ACS and PACS are used to understand time series, fit the appropriate models, and make the forecast. The term correlation is linked to the lags, which is nothing else than the correlation for a lag between two observations. For example, the lag between the previous and the current observation is lag=1. So when looking at ACS, if the values are near zero, it means no correlation, does not have a trend or seasonal pattern, and therefore we are in the presence of a stationary time series.

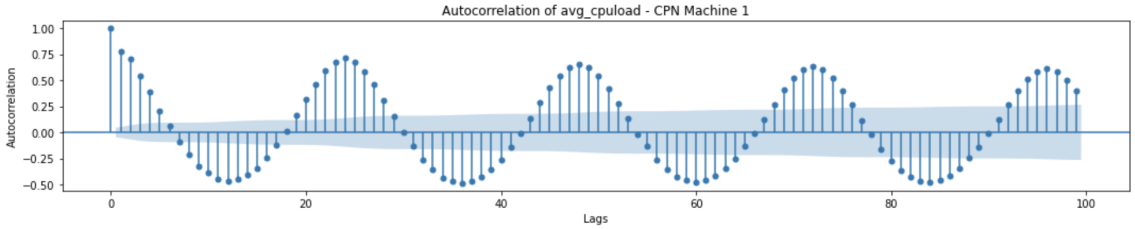


Figure 5.12: Autocorrelation of avg_cpuload CPN Machine 1

When there are seasonal patterns, the autocorrelations are higher for multiples of the seasonal frequency than for other lags. When there is both trend and seasonality, the ACS displays a mixture of both effects.

In the figure Figure 5.12, we can see a trend since the correlations decrease slowly to zero. At the same time, it increases for every multiple of 24, meaning there is seasonality. First, a simple differencing and trend removal is done.

In the figure Figure 5.13 we can see that there is no more trend since almost all values are near zero except for lags=24,48,72. A differencing of S=24 and seasonality removal is done.

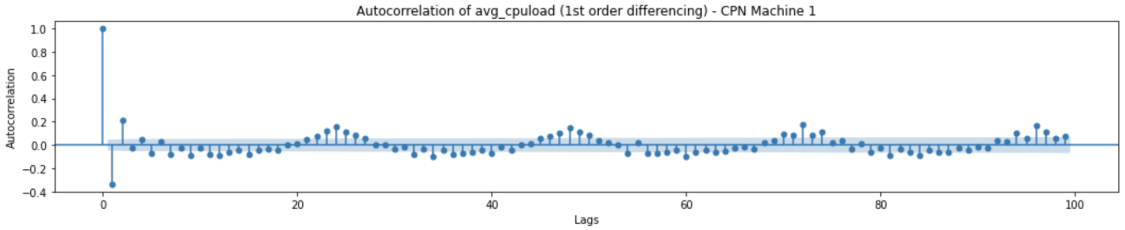


Figure 5.13: Autocorrelation of avg_cpuload (1st order Differencing) CPN Machine 1

With seasonal differencing, time series seems to become stationary. We can use Augmented Dickey–Fuller (ADF) test to test this assumption. ADF tests the null hypothesis that a unit root is present in a time series sample. In this test, the

Figure 5.14: Autocorrelation of avg_cpuload (1st order Differencing & Seasonal Differencing) CPN Machine 1

Figure 5.15: Partial Autocorrelation of avg_cpuload (1st order Differencing & Seasonal Differencing) CPN Machine 1

negative number is used, and the more negative the number is, the stronger the rejection of the hypothesis that there is a unit root at some confidence level. More can be found at [32].

```
ADF Statistic: -12.892347
p-value: 0.000000
Critical Values:
        1%: -3.434
        5%: -2.863
        10%: -2.568
```

Figure 5.16: ADF Test of the avg_cpuload without trend and seasonality CPN Machine 1

In the Figure 5.16 we can see that the p_values is zero and the ADF statistic is lower than the critical values at 1%. Meaning that first-order seasonal differencing seems adequate, so D=1.

Now on, we have two options, obtain p,q, P, and Q by interpreting ACS or PACS or performing a grid-search and saving the parameters, which led to forecasting with lower errors.

First, the parameters were obtained by manual interpretation. Looking ACS in Figure 5.14 and PACS in Figure 5.15 for lag in the range of 0-24 we can observe that there are some values falling out of the confidence bound, so p and q can assume multiple values, but at lag=1 we obtain a simpler model. For now, p=1 and q=1 are estimated. To get P and Q we need to observe ACS and PACS for the lags=[0,24,48,72].

Looking at Figure 5.17 we can see that the value out of the confidence bounds is the first one at lag=24, so P=1. When looking at Figure 5.18, we can see that several values are out of the confidence bounds, but a lag=24 is chosen, which

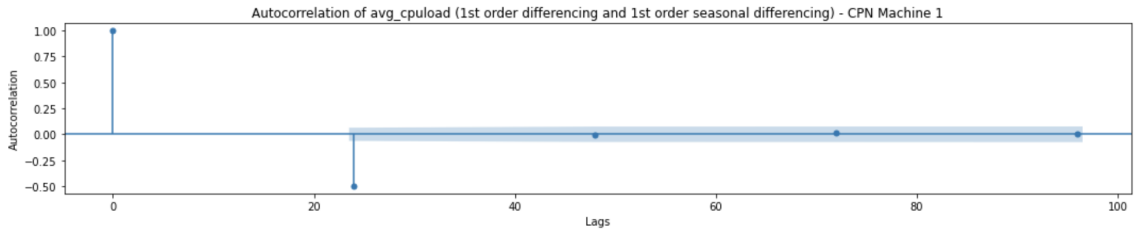


Figure 5.17: Re-sampled Autocorrelation of avg_cpuload (1st order Differencing & Seasonal Differencing) CPN Machine 1

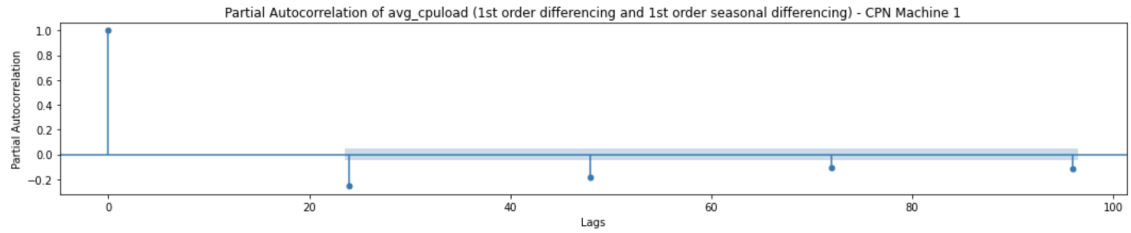


Figure 5.18: Re-sampled Partial Autocorrelation of avg_cpuload (1st order Differencing & Seasonal Differencing) CPN Machine 1

corresponds to Q=1 since it is the simpler model.

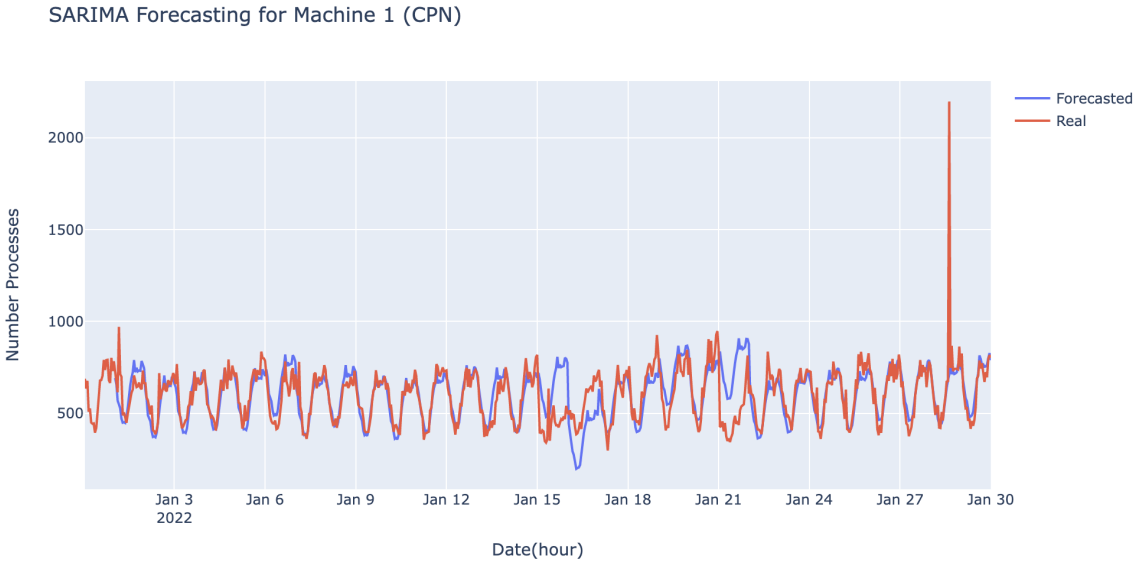


Figure 5.19: SARIMA Forecasting January for Machine 1 CPN

In the figure Figure 5.19 we have the result of using the parameters studied before, meaning p=1,d=1, q=1, P=1, D=1, Q=1, S=24. When applying forecasting for unknown values, it results in 12.2% MAPE. Compared to LSTM also during the validation step, which resulted in 13.4% for univariate data, we have an improvement of 1.2%. However, compared to LSTM for multivariate data, it is worse than 0.8%. Therefore, using its statistical properties, tendency, and seasonal part adds crucial information when forecasting at the validation step.

The decisions based on ACS and PACS may not lead to the best possible parameters for the available data. We can use grid-search to fine-tune the models and get the parameters with a lower error when forecasting unknown values.

The model was trained and tested for the grid search with the sets described before, saving the result with the lowest MSE. The testing parameters are:

- p = [0,1,2]
- q = [0,1,2]
- P = [0,1,2]
- Q = [0,1,2]

The parameters with the lowest MSE for the train validation part are in the table Table 5.4 for CPN, in the table Table 5.5 for DSGW and in the table Table 5.6 for SMP.

In Table 5.4, we can see that machine one already improves the result obtained by manually analyzing the correlation. Let us see how it behaves along the data and compare it with the LSTM models before.

SARIMA Forecasting best parameters for all machines of CPN								
Machine	MAPE	p	d	q	P	D	Q	Seasonality
1	12.01	1	1	1	0	1	1	24
2	11.84	0	1	1	0	1	1	24
3	14.26	0	1	1	0	1	1	24
4	13.81	0	1	1	0	1	1	24
5	14.55	0	1	1	0	1	1	24
6	13.48	0	1	1	0	1	1	24
7	12.7	0	1	1	1	1	1	24
8	14.16	0	1	1	0	1	1	24
9	23.2	1	1	1	0	1	1	24
10	18.6	1	1	1	1	1	1	24
11	19.02	0	1	1	1	1	1	24
12	16.82	0	1	1	1	1	1	24
13	17.16	0	1	1	1	1	1	24
14	20.16	0	1	1	0	1	1	24
15	16.29	1	1	1	1	1	1	24
16	17.77	0	1	1	0	1	1	24

Table 5.4: Best SARIMA parameters for all machines of CPN: training with data from October until December and forecasting January

SARIMA Forecasting best parameters for all machines of DSGW								
Machine	MAPE	p	d	q	P	D	Q	Seasonality
1	36.43	0	1	0	0	1	1	24
2	15.75	0	1	0	0	1	1	24
3	20.71	1	1	1	1	1	1	24
4	22.01	0	1	1	0	1	1	24

Table 5.5: Best SARIMA parameters for all machines of DSGW: training data from October until December and forecasting January

SARIMA Forecasting best parameters for all machines of SMP								
Machine	MAPE	p	d	q	P	D	Q	Seasonality
1	9.04	1	1	1	0	1	1	24
2	9.61	0	1	1	0	1	1	24
3	9.52	0	1	1	1	1	1	24
4	10.55	1	1	1	0	1	1	24
5	11.17	0	1	1	1	1	1	24
6	10.76	0	1	1	0	1	1	24

Table 5.6: Best SARIMA parameters for all machines of SMP: training data from October until December and forecasting January

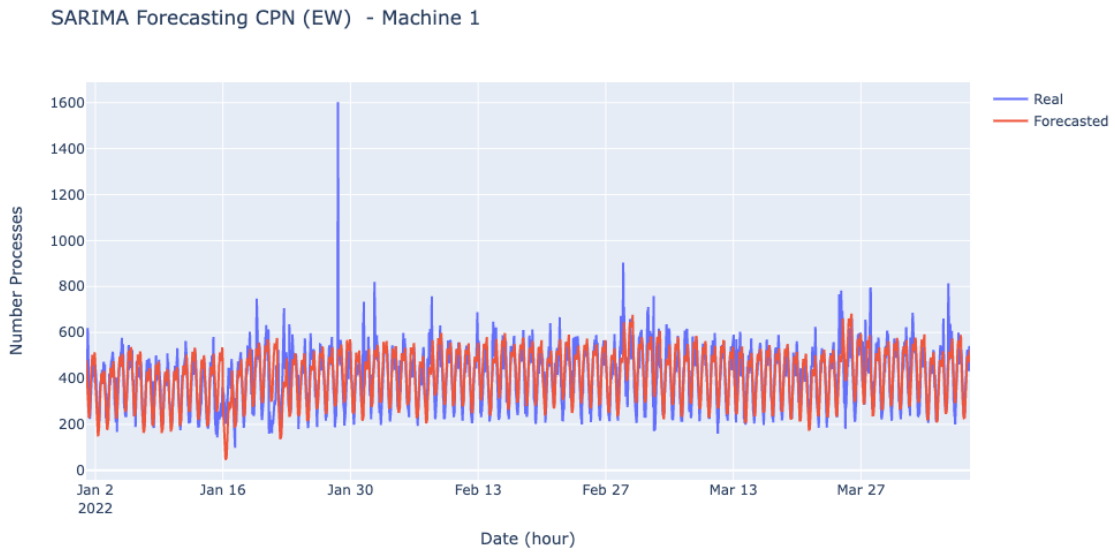


Figure 5.20: SARIMA Forecasting for Machine 1 CPN - Expanding Window to forecast data from January until April

In the Figure 5.20 we can see how it behaves visually and in the table Table 5.10 we can see it more specifically. When it comes to machine 1, with an expanding window, there is an improvement than with LSTM, leading to an overall error of 13.71%. When looking at the outliers points from those above the lowest threshold, there is a MAPE of 25.2%, above the medium threshold is 29.48% and for the highest threshold is 38.41%. As well as the overall error, when it comes to the MAPE at the anomaly points SARIMA performs comparing forecasting with

Different Train Size Sliding Window for SARIMA - CPN				
Machine	Size=48	Size=72	Size=168	Size=720
1	46.46	23.14	18.88	17.6
2	19.37	16.6	14.01	12.36
3	22.81	19.59	16.96	14.62
4	21.24	17.29	14.82	13.97
5	23.47	20.41	15.91	14.88
6	20.19	18.24	15.42	13.99
7	20.06	17.66	14.8	12.93
8	21.75	17.46	16.5	14.37
9	59.88	34.31	28.43	23.38
10	30.48	22.91	20.78	18.74
11	24.36	30.55	20.35	18.69
12	128.75	22.48	17.71	17.08
13	123.71	24.89	19.4	17.36
14	145.66	24.85	21.72	20.05
15	118.86	24.07	17.76	16.42
16	108.95	21.39	19.66	17.86

Table 5.7: Grid Search Result for Different Training Size SARIMA Sliding Window (CPN) forecating January: higher values colored in red and lower values colored in green each size

LSTM for both univariate data and multivariate data. The same applies to almost all machines for all sites, SARIMA expanding window results in better results. Even if it results in better results, in the Table 5.25 we can see that SARIMA takes much more time than the other models with almost 30 seconds. It may look like a small amount of time, but because we are using an expanding window, this number will increase when increasing data daily.

The increment of the training can be overcome by using a fixed training size. The parameters from the expanding window approach are maintained, and a grid search is done to test 4 different training sizes two days, three days, a week, and a month. The training set cannot have a higher than a month size because of the dataset size. A sliding window for different sizes is made to compare its performance with the ones obtained from the expanding window.

We can see in the Table 5.7 for the site CPN, Table 5.8 for site DSGW and Table 5.9 for site SMP that for almost all the machines the training size with lower MAPE is the highest, which is size of 720 and as expected using only two days doesn't

Different Train Size Sliding Window for SARIMA - DSGW				
Machine	Size=48	Size=72	Size=168	Size=720
1	20.62	22.05	21.52	20.96
2	24.12	22.09	19.67	17.55
3	23.67	18.89	17.18	16.65
4	24.0	19.41	16.91	16.47

Table 5.8: Grid Search Result for Different Training Size SARIMA Sliding Window (DSGW) forecasting January: higher values colored in red and lower values colored in green each size

Different Train Size Sliding Window for SARIMA - SMP				
Machine	Size=48	Size=72	Size=168	Size=720
1	20.62	22.05	21.52	20.96
2	24.12	22.09	19.67	17.55
3	23.67	18.89	17.18	16.65
4	24.0	19.41	16.91	16.47

Table 5.9: Grid Search Result for Different Training Size SARIMA Sliding Window (SMP) forecasting January: higher values colored in red and lower values colored in green each size

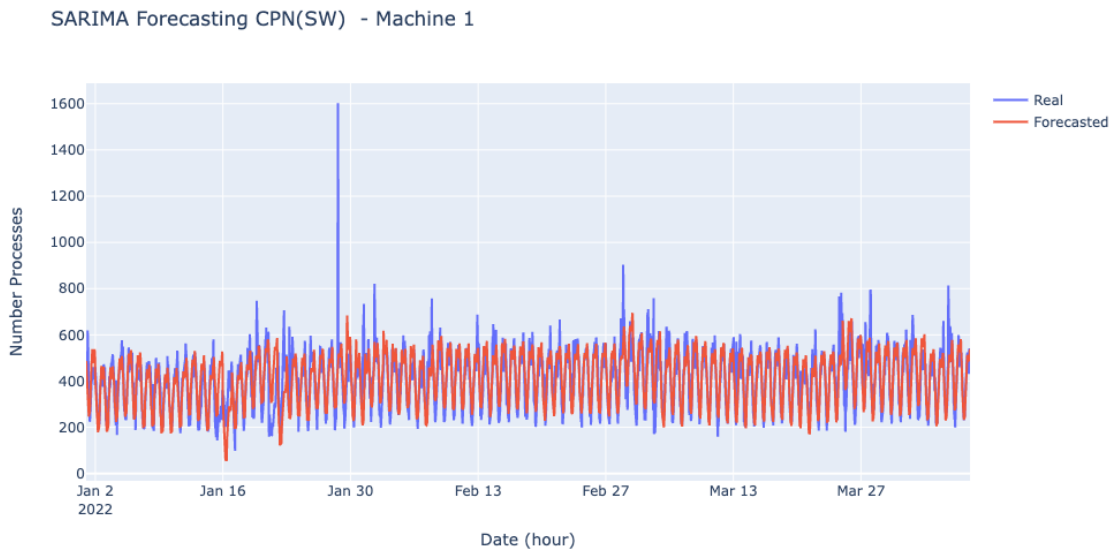


Figure 5.21: SARIMA Forecasting for Machine 1 CPN - Sliding Window to forecast data from January until April

result in the best results. The final step is forecasting with SARIMA using the sliding window of the size with the lowest error works for the full dataset.

In the Figure 5.21, we can see how forecasting using a sliding window of 720, which corresponds to 1-month data training data, behaves during all data. Compared to expanding window, the overall MAPE decreased to 13.24%. When looking at the anomaly error, it decreased to 23.61% above the lower threshold, to

23.61% above the medium threshold, and when it comes to the values above the highest, it increased a little bit to 38.61%. When we look at the forecasted data without outliers, it is lower for all thresholds, 11.85% above the lowest, 12.55% above the medium, and 12.88% above the highest.

When looking at the other machines for all the sites in table Table 5.10 for CPN, Table 5.11 for DSGW and Table 5.12 for SMP, forecasting using SARIMA and sliding window approach resulted for almost all of them in the lowest error. The one with a different size resulted in a higher error than the expanding window. Besides the improvement on the MAPE, there is also an improvement in the training time which is 5.61 seconds. In addition to an improvement of 18%, this value is fixed as the training size does not change with time.

SARIMA Forecasting MAPE (OvE), (OtE), (NOE) CPN									
Mch.	Type W.	Size W.	OvE	Outliers Error			Non Outliers Error		
				k=1.5	k=1.75	k=2	k=1.5	k=1.75	k=2
1	EW	-	13.71	25.2	29.48	38.41	12.17	13.01	13.36
	SW	720	13.24	23.61	28.62	38.61	11.85	12.55	12.88
2	EW	-	8.67	12.07	16.41	21.76	8.27	8.43	8.56
	SW	720	8.57	11.42	16.46	22.57	8.23	8.32	8.44
3	EW	-	10.32	15.82	23.45	30.09	9.63	9.95	10.18
	SW	720	10.12	15.02	23.05	30.74	9.52	9.76	9.98
4	EW	-	10.18	16.39	20.23	28.57	9.44	9.83	10.03
	SW	720	9.84	15.56	19.23	28.88	9.16	9.52	9.68
5	EW	-	10.15	16.31	23.59	32.16	9.48	9.77	9.96
	SW	720	9.86	15.83	23.37	31.74	9.2	9.47	9.67
6	EW	-	10.07	14.52	17.3	40.63	9.55	9.85	9.96
	SW	720	9.83	13.47	17.16	40.81	9.41	9.61	9.72
7	EW	-	9.33	12.79	17.5	29.68	8.87	9.11	9.25
	SW	720	8.93	12.18	17.31	32.13	8.5	8.7	8.84
8	EW	-	10.12	15.49	19.54	37.2	9.46	9.83	9.96
	SW	720	9.87	14.41	19.17	37.64	9.31	9.58	9.7
9	EW	-	12.88	21.37	32.35	44.64	11.8	12.15	12.42
	SW	720	13.19	20.35	31.49	44.8	12.28	12.5	12.73
10	EW	-	12.37	20.53	25.55	34.53	11.37	11.89	12.02
	SW	720	12.37	20.45	26.55	37.57	11.38	11.86	11.98
11	EW	-	11.58	18.18	25.27	41.52	10.69	11.05	11.24
	SW	720	11.35	17.65	24.59	40.43	10.5	10.84	11.03
12	EW	-	11.2	19.57	32.23	43.98	10.02	10.4	10.73
	SW	720	10.95	19.64	32.15	43.79	9.72	10.14	10.48
13	EW	-	10.96	18.11	26.29	46.04	10.03	10.3	10.52
	SW	720	10.89	18.34	26.6	46.1	9.92	10.21	10.45
14	EW	-	11.82	18.55	28.98	58.32	10.9	11.13	11.35
	SW	720	11.51	18.43	29.08	58.79	10.56	10.81	11.04
15	EW	-	10.53	15.62	22.27	33.86	9.89	10.12	10.27
	SW	720	10.28	15.43	22.37	33.81	9.63	9.86	10.02
16	EW	-	11.18	16.02	23.22	cellcolorverde 42.21	10.54	10.73	10.87
	SW	720	10.95	16.26	23.84	42.97	10.25	10.46	10.63

Table 5.10: SARIMA Forecasting CPN between January and April (OvE) overall Error, (OtE) outliers error for each threshold and (NOE) non outliers error for the lowest threshold: for each type of window (Type W.) which can be sliding window(SW) or expanding window (EW). For each machine, the lowest MAPE between expanding window and sliding window are colored int green

SARIMA Forecasting MAPE (OvE), OtE), (NOE) DSGW									
Mch.	Type W.	Size W.	OE	Outliers Error			Non Outliers Error		
				k=1.5	k=1.75	k=2	k=1.5	k=1.75	k=2
1	EW	-	13.02	28.42	47.49	48.91	12.14	12.74	12.9
	SW	48	15.3	27.23	48.73	50.61	14.62	15.03	15.18
2	EW	-	11.8	20.58	34.79	51.86	11.39	11.66	11.68
	SW	720	11.91	20.61	35.5	52.52	11.5	11.77	11.79
3	EW	-	10.46	20.0	64.83	44.78	9.86	10.06	10.37
	SW	720	9.92	18.73	62.61	42.99	9.36	9.53	9.83
4	EW	-	10.13	22.38	67.35	63.72	9.33	9.78	9.94
	SW	720	10.06	21.8	68.87	64.52	9.29	9.7	9.87

Table 5.11: SARIMA Forecasting DSGW between January and April (OvE) overall Error, (OtE) outliers error for each threshold and (NOE) non outliers error for the lowest threshold: for each type of window (Type W.) which can be sliding window(SW) or expanding window (EW). For each machine, the lowest MAPE between expanding window and sliding window are colored int green

SARIMA Forecasting MAPE (OvE), (OtE), (NOE) SMP									
Mch.	Type W.	Size W.	OvE	Outliers Error			Non Outliers Error		
				k=1.5	k=1.75	k=2	k=1.5	k=1.75	k=2
1	EW	-	7.49	9.48	9.69	18.17	7.17	7.31	7.27
	SW	720	7.9	9.81	9.86	19.21	7.59	7.74	7.66
2	EW	-	8.82	12.38	12.79	23.96	8.21	8.51	8.56
	SW	720	8.94	12.17	12.58	23.4	8.39	8.65	8.69
3	EW	-	8.99	12.65	12.25	22.34	8.39	8.7	8.67
	SW	720	9.16	12.98	12.61	22.14	8.53	8.86	8.85
4	EW	-	9.64	24.65	30.73	42.04	7.68	8.24	8.55
	SW	720	9.85	23.78	30.33	42.06	8.04	8.49	8.77
5	EW	-	10.7	24.31	33.41	45.05	9.0	9.44	9.74
	SW	720	10.25	22.49	31.19	42.14	8.72	9.09	9.36
6	EW	-	10.81	24.31	31.31	42.1	9.05	9.53	9.72
	SW	720	10.53	22.95	29.77	40.28	8.91	9.33	9.49

Table 5.12: SARIMA Forecasting SMP between January and April (OvE) overall Error, (OtE) outliers error for each threshold and (NOE) non outliers error for the lowest threshold: for each type of window (Type W.) which can be sliding window(SW) or expanding window (EW). For each machine, the lowest MAPE between expanding window and sliding window are colored int green

5.3.2 TES Results

In the case of TES, as it is the most complex exponential smoothing model, it deals with level, trend, and seasonality as explained on section 2.3. In addition to these parameters varying from 0 to 1, trend or seasonality can be multiplicative or additive. When looking at the time series, it cannot detect if the trend is multiplicative or additive as it changes with time. The goal, for now, is to obtain fixed model parameters and update them with new data, whereas in the future may be helpful to implement dynamic model updates with time. The grid search aims to get a general model based on historical data forecast with the lowest error unseen data. The parameters tested using an expanding window during grid-search were:

- trend = ['add','mul']
- seasonality = ['add','mul']
- period = 24
- alpha = np.arange(0, 1, 0.1)
- beta = np.arange(0, 1, 0.1)
- gamma = np.arange(0, 1, 0.1)

The best solution for machine one at site CPN using an expanding window had the following parameters:

- trend = 'add'
- seasonality = 'add'
- period = 24
- alpha = 0.3
- beta = 0.7
- gamma = 0.8

We can see the parameters of the other machines at site CPN on Table 5.13, at site DSGW on Table 5.14 and at the site Table 5.15. It is clear that for the type of trend and seasonality, it is additive for all machines at all sites. There is no clear generalization regarding the decreasing rate of the level, trend, and seasonality. As explained in section 2.3, higher values make the model pay attention mainly to the most recent past values. Looking at machine one, we can see that only the most recent one counts regarding seasonality. For the trend and importance, they both have the same lower values meaning that it looks further into the past.

TES Forecasting best parameters for all machines of CPN							
Machine	MAPE	Trend	Seasonal	Seasonal Period	Alpha	Beta	Gamma
1	19.6	add	add	24	0.4	0.4	0.9
2	11.24	add	add	24	0.4	0.9	0.1
3	14.7	add	add	24	0.2	0.9	0.5
4	14.19	add	add	24	0.7	0.4	0.9
5	14.43	add	add	24	0.6	0.6	0.6
6	12.95	add	add	24	0.8	0.5	0.5
7	13.15	add	add	24	0.1	0.0	0.3
8	14.78	add	add	24	0.3	0.5	0.1
9	24.68	add	add	24	0.1	0.5	0.1
10	19.78	add	add	24	0.6	0.6	0.8
11	19.07	add	add	24	0.4	0.4	0.2
12	18.54	add	add	24	0.2	0.0	0.5
13	18.73	add	add	24	0.5	0.5	0.5
14	20.59	add	add	24	0.3	0.7	0.9
15	17.34	add	add	24	0.5	0.7	0.9
16	19.47	add	add	24	0.3	0.5	0.3

Table 5.13: Best TES parameters for all machines of CPN: training with data from October until December and forecasting January

TES Forecasting best parameters for all machines of DSGW							
Machine	MAPE	Trend	Seasonal	Seasonal Period	Alpha	Beta	Gamma
1	28.59	add	add	24	0.4	0.3	0.0
2	22.48	add	add	24	0.4	0.0	0.7
3	21.74	add	add	24	0.2	0.7	0.6
4	18.27	add	add	24	0.6	0.1	0.3

Table 5.14: Best TES parameters for all machines of DSGW: using data from October until December and forecasting January

TES Forecasting best parameters for all machines of SMP							
Machine	MAPE	Trend	Seasonal	Seasonal Period	Alpha	Beta	Gamma
1	12.32	add	add	24	0.7	0.9	0.0
2	13.11	add	add	24	0.0	0.2	0.3
3	12.67	add	add	24	0.6	0.2	0.1
4	14.23	add	add	24	0.0	0.6	0.5
5	14.14	add	add	24	0.4	0.2	0.3
6	13.75	add	add	24	0.4	0.6	0.3

Table 5.15: Best TES parameters for all machines of SMP: using data from October until December and forecasting January

This resulted in an error of 19.6%, which is pretty high compared with the previous testing step of the solutions. We can see the forecasting result in the figure Figure 5.22

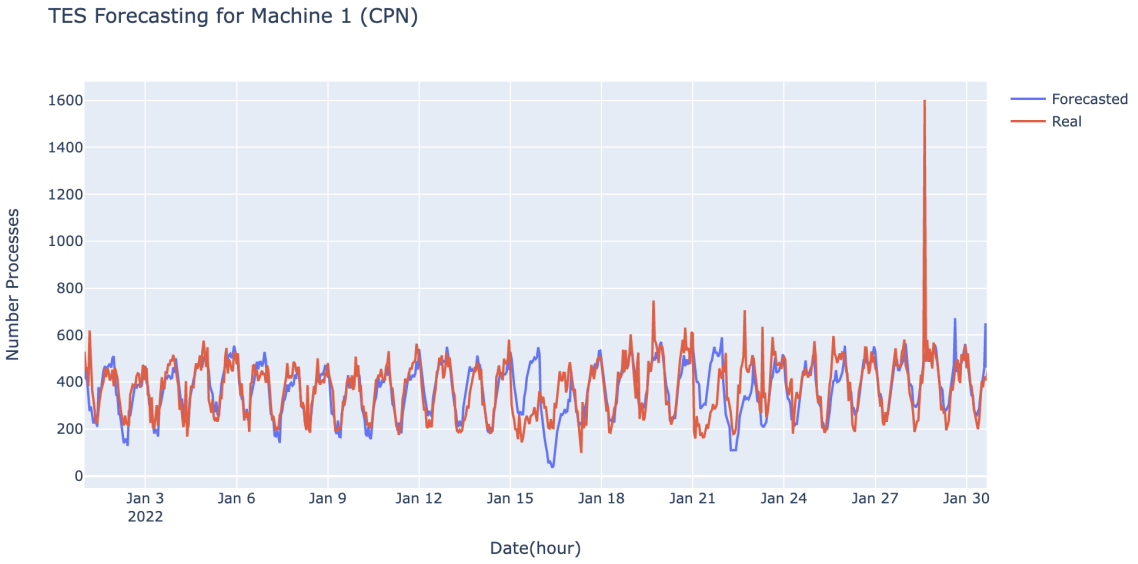


Figure 5.22: TES Forecasting for Machine 1 CPN - Forecasting January

Now that we have the best TES models let us see how an expanding window behaves along the data while forecasting anomaly points in the figure Figure 5.23 starting from January.

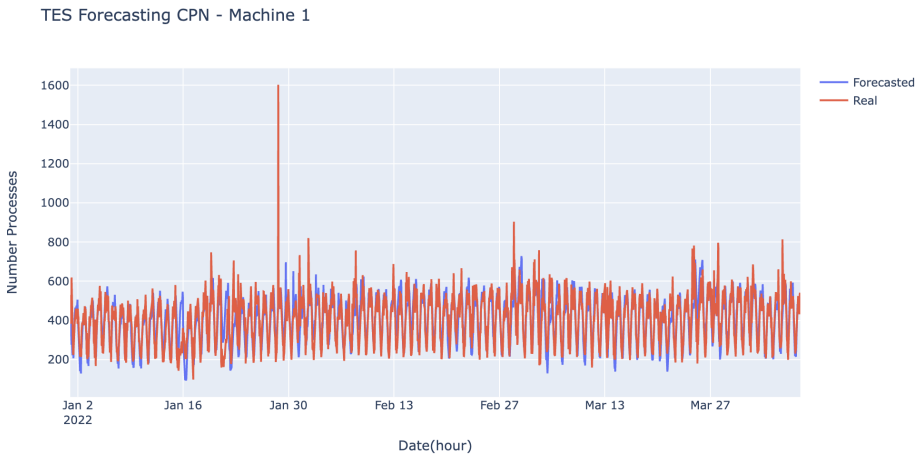


Figure 5.23: TES Forecasting for Machine 1 CPN - Expanding Window forecasting from January until April

We get an overall MAPE of 14.2% which compared to LSTM it is lower but when compared to SARIMA it is higher. As for the outliers error above, the lowest MAPE is 24.21%, above the medium 28.2%, and the highest is 37.58%. If we look

at the forecasting result without outliers, the values 12.86% remove the values above the lowest threshold, 13.58% above the medium threshold, and 13.87% above the highest threshold. We can see that MAPE without anomalies get lower than the overall error, meaning that our model almost always fails by a significant difference compared to the real values.

Besides the fact that using TES with expanding window still results in a high error and because our train data is constantly expanding with time to even so as can be seen in the Table 5.25 TES is the best one by far when it comes to the training time.

We can overcome this issue by using a sliding window, meaning a fixed size. As it was done before, we are going to be tested in four different sizes: two days, three days, a week, and a month. We cannot use more than a month for our test because of our data size.

Different Train Size Sliding Window for TES - DSGW				
Machine	Size=48	Size=72	Size=168	Size=720
1	28.51	23.36	25.5	26.22
2	27.14	25.41	23.25	22.33
3	20.0	19.07	17.31	17.93
4	23.41	20.49	16.39	20.59

Table 5.16: Grid Search Result for Different Training Size TES Sliding Window (DSGW) forecasting January: higher values colored in red and lower values colored in green each size

Different Train Size Sliding Window for TES - SMP				
Machine	Size=48	Size=72	Size=168	Size=720
1	18.64	13.98	13.43	18.37
2	25.77	23.73	13.8	16.83
3	22.2	21.06	14.49	16.89
4	21.73	17.58	15.96	21.4
5	22.18	17.75	14.76	19.39
6	18.31	16.81	14.88	16.59

Table 5.17: Grid Search Result for Different Training Size TES Sliding Window (SMP) forecasting January: higher values colored in red and lower values colored in green each size

Different Train Size Sliding Window for TES - CPN				
Machine	Size=48	Size=72	Size=168	Size=720
1	24.61	23.71	26.45	19.39
2	18.52	21.86	21.9	21.75
3	21.18	20.58	25.45	22.78
4	19.77	18.53	22.73	19.58
5	22.09	20.5	26.11	24.41
6	20.39	19.99	21.75	21.43
7	18.77	16.8	24.76	18.39
8	18.95	18.12	24.3	20.39
9	31.75	31.55	37.52	33.76
10	23.15	23.22	25.35	28.31
11	22.06	19.45	26.21	31.48
12	29.1	23.91	28.27	24.61
13	28.37	26.21	28.16	25.82
14	30.38	28.55	32.09	26.36
15	24.15	25.47	26.43	24.61
16	22.58	22.08	26.07	31.2

Table 5.18: Grid Search Result for Different Training Size TES Sliding Window (CPN) forecasting January: higher values colored in red and lower values colored in green each size

Looking at the Table 5.18 for machine one, we can see that as for SARIMA sliding window, the one with the lowest error is using 720 values, which corresponds to 1 month. When it comes to the other machines for the other sites, we cannot make a general conclusion besides that for the CPN site, the size with higher error is when using 168 values, and the size with the highest number of low errors is when using 72 values. For DSGW, the size with the highest errors is when forecasting using only 48h, and we cannot make a straightforward generalization for the size with the lowest errors. When it comes to SMP, there is an apparent size with the highest error, which is when training for 48 hours, and the size with the lowest error are when using 168 hours to train the model.

Let us see how the best model per machine performs using TES with the sliding window approach. The goal of testing a smaller training size is to save time when forecasting and verify if the historical data matters.

In the Figure 5.24, we can see forecasting using a training size of a month for machine 1 of site CPN. The error values can be conferred in Table 5.19 where it can be seen that even using the largest training size results in a higher value of 16.31% compared to the expanding window. The error for the outliers values increases compared to the overall error, and it is also higher compared to the anomaly error using expanding window reaching almost 50% of MAPE. The values without anomalies are lower than the overall error, which makes sense since the anomaly error is so high. We can take the same conclusions regarding the rest of the ma-

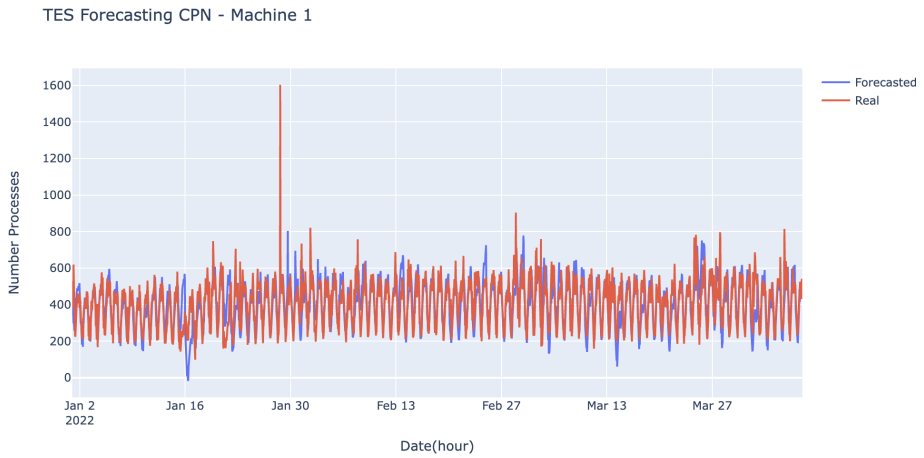


Figure 5.24: TES Forecasting for Machine 1 CPN - Sliding Window forecasting from January until April

chines at CPN, as seen on Table 5.19. As for DSGW on Table 5.20 and SMP on Table 5.21 sliding window is the one with lowest errors even so not as much the result from the previous models as can be seen.

TES Forecasting MAPE (OvE), (OtE), (NOE) CPN									
Mch.	Type W.	Size W.	OE	Outliers Error			Non Outliers Error		
				k=1.5	k=1.75	k=2	k=1.5	k=1.75	k=2
1	EW	-	14.2	24.21	28.2	37.58	12.86	13.58	13.87
	SW	720	16.31	27.21	31.65	40.15	14.85	15.63	15.98
2	EW	-	10.01	13.57	18.73	24.58	9.59	9.74	9.89
	SW	48	13.33	17.65	20.55	26.93	12.82	13.11	13.22
3	EW	-	11.42	17.21	25.99	35.88	10.7	11.01	11.25
	SW	72	16.22	22.54	28.95	39.09	15.43	15.86	16.06
4	EW	-	10.46	15.67	20.29	30.72	9.84	10.12	10.3
	SW	72	14.61	20.11	25.44	38.63	13.96	14.24	14.41
5	EW	-	11.81	18.45	25.82	35.34	11.08	11.41	11.6
	SW	72	14.12	21.36	28.77	35.07	13.33	13.7	13.93
6	EW	-	10.82	14.61	16.87	39.51	10.38	10.64	10.72
	SW	72	14.43	19.8	25.87	46.99	13.8	14.09	14.32
7	EW	-	10.22	13.83	19.35	36.29	9.75	9.98	10.12
	SW	72	14.29	19.05	23.78	40.72	13.67	14.04	14.19
8	EW	-	10.89	15.57	21.11	34.88	10.3	10.57	10.74
	SW	72	15.12	24.95	31.11	41.94	13.9	14.62	14.95
9	EW	-	15.52	23.3	32.18	41.28	14.54	14.9	15.15
	SW	72	17.95	23.89	36.02	53.7	17.19	17.27	17.43
10	EW	-	13.45	20.99	25.33	35.42	12.52	13.02	13.11
	SW	48	18.55	23.34	28.74	39.94	17.96	18.18	18.22
11	EW	-	12.46	18.62	25.23	41.09	11.63	11.97	12.14
	SW	72	15.31	21.19	23.23	35.8	14.52	15.0	15.08
12	EW	-	12.38	21.38	34.7	46.72	11.11	11.53	11.89
	SW	72	15.4	23.78	37.15	53.16	14.22	14.57	14.86
13	EW	-	12.33	20.25	28.35	46.0	11.3	11.64	11.91
	SW	720	15.75	24.09	31.58	51.01	14.66	15.06	15.3
14	EW	-	13.32	21.4	32.5	63.98	12.21	12.55	12.81
	SW	720	16.34	24.91	33.65	62.96	15.17	15.65	15.88
15	EW	-	12.35	19.29	27.51	41.49	11.48	11.83	12.03
	SW	48	14.89	20.97	27.57	41.49	14.13	14.45	14.6
16	EW	-	12.62	19.26	27.14	45.32	11.74	12.07	12.29
	SW	72	15.01	21.65	27.61	49.37	14.14	14.54	14.67

Table 5.19: TES Forecasting CPN between January and April (OvE) overall Error, (OtE) outliers error for each threshold and (NOE) non outliers error for the lowest threshold: for each type of window (Type W.) which can be sliding window(SW) or expanding window (EW). For each machine, the lowest MAPE between expanding window and sliding window are colored int green

TES Forecasting MAPE (OvE), (OtE), (NOE) DSGW									
Mch.	Type W.	Size W.	OE	Outliers Error			Non Outliers Error		
				k=1.5	k=1.75	k=2	k=1.5	k=1.75	k=2
1	EW	-	16.45	33.63	49.24	50.95	15.47	16.18	16.33
	SW	72	14.13	26.54	45.9	49.49	13.42	13.87	14.01
2	EW	-	18.99	28.64	35.47	46.76	18.54	18.89	18.91
	SW	720	16.43	24.72	33.1	49.91	16.04	16.33	16.33
3	EW	-	14.84	30.43	88.44	64.0	13.86	14.3	14.71
	SW	168	10.99	20.53	65.69	43.42	10.39	10.58	10.9
4	EW	-	13.04	25.89	70.87	69.91	12.2	12.69	12.84
	SW	168	10.31	22.5	69.98	65.06	9.52	9.95	10.12

Table 5.20: TES Forecasting DSGW between January and April (OvE) overall Error, (OtE) outliers error for each threshold and (NOE) non outliers error for the lowest threshold: for each type of window (Type W.) which can be sliding window(SW) or expanding window (EW). For each machine, the lowest MAPE between expanding window and sliding window are colored int green

TES Forecasting MAPE (OvE), (OtE), (NOE) SMP									
Mch.	Type W.	Size W.	OE	Outliers Error			Non Outliers Error		
				k=1.5	k=1.75	k=2	k=1.5	k=1.75	k=2
1	EW	-	11.6	15.27	14.57	22.65	11.01	11.36	11.37
	SW	168	9.24	11.53	11.43	19.44	8.87	9.06	9.03
2	EW	-	11.25	15.84	15.87	24.98	10.47	10.89	11.02
	SW	168	9.5	12.35	12.85	22.37	9.01	9.23	9.28
3	EW	-	11.39	15.82	15.66	23.66	10.67	11.02	11.1
	SW	168	9.83	14.0	14.1	22.45	9.15	9.47	9.53
4	EW	-	14.84	30.0	34.72	43.8	12.86	13.52	13.87
	SW	168	10.72	24.8	31.51	44.09	8.88	9.34	9.6
5	EW	-	13.76	27.88	36.49	47.0	12.0	12.5	12.83
	SW	168	10.36	22.92	31.35	42.22	8.8	9.2	9.47
6	EW	-	14.67	28.09	35.89	44.21	12.92	13.35	13.65
	SW	168	10.88	23.74	30.87	40.89	9.21	9.64	9.84

Table 5.21: TES Forecasting SMP between January and April (OvE) overall Error, (OtE) Outliers error for each threshold and (NOE) non outliers error for the lowest threshold: for each type of window (Type W.) which can be sliding window(SW) or expanding window (EW). For each machine, the lowest MAPE between expanding window and sliding window are colored int green

5.4 Data Visualization

After getting the best models with the lowest error, as shown in the previous sections, the final forecast evolving in time, and business data progression can be uploaded to the visualization tool.

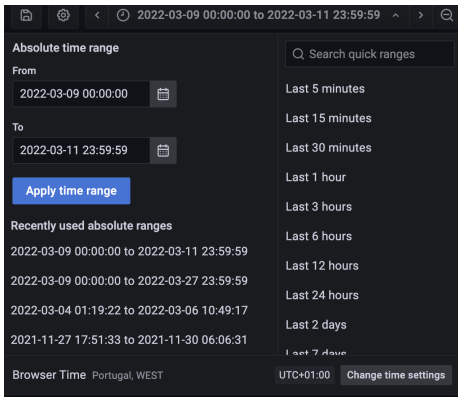


Figure 5.26: Time Range Data Visualization

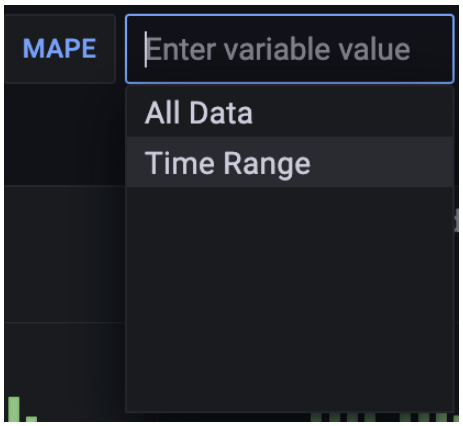


Figure 5.27: Metric Options

In the Figure 5.25, we can see the dashboard for two days starting on the 9th of March of 2022 that can be chosen from a pop-up as shown in Figure 5.26.

Furthermore, below the title of the dashboard and time range can be selected a machine for CPN, DSGW, and SMP from a dropdown list as sown in Figure 5.28,Figure 5.29 and Figure 5.30 as well as choosing the metric time range error type from a dropdown as shown in Figure 5.27.

In the Figure 5.28,Figure 5.29 and Figure 5.30, we can see all machine options from each site with their real names. When selecting a machine, the forecasting using TES and SARIMA expanding and sliding window, as well as LSTM using univariate and multivariate data. We can select a specific time series in the time chart from which we see the forecasts and actual values, as shown in Figure 5.31. Each forecast has a specific color with a darker color for its complexity. TES color is purple in the metric boxes and in the time chart, while the expanding window approach acquires a darker tone. For SARIMA, the color is blue, and the darker blue represents expanding window. Finally, LSTM is presented by the color red, whereas a darker red shows the multivariate approach.

To get a clearer view of the performance for each type of forecasting, we can choose how to view the metric, in this case, MAPE. We can choose to get the error for a specific "Time Range" in this case for two days starting on the 9th of March, or we can choose the "All Data" option, where MAPE will be calculated for all the forecasting values. In each metric box per model besides the metric, we can see how the error behaves during the time range selected or for all data depending on the chosen metric option.

On the right of the forecasting timeline and metrics are the graphs of the business request represented in the timeline graph colored in green. In gray, there is a

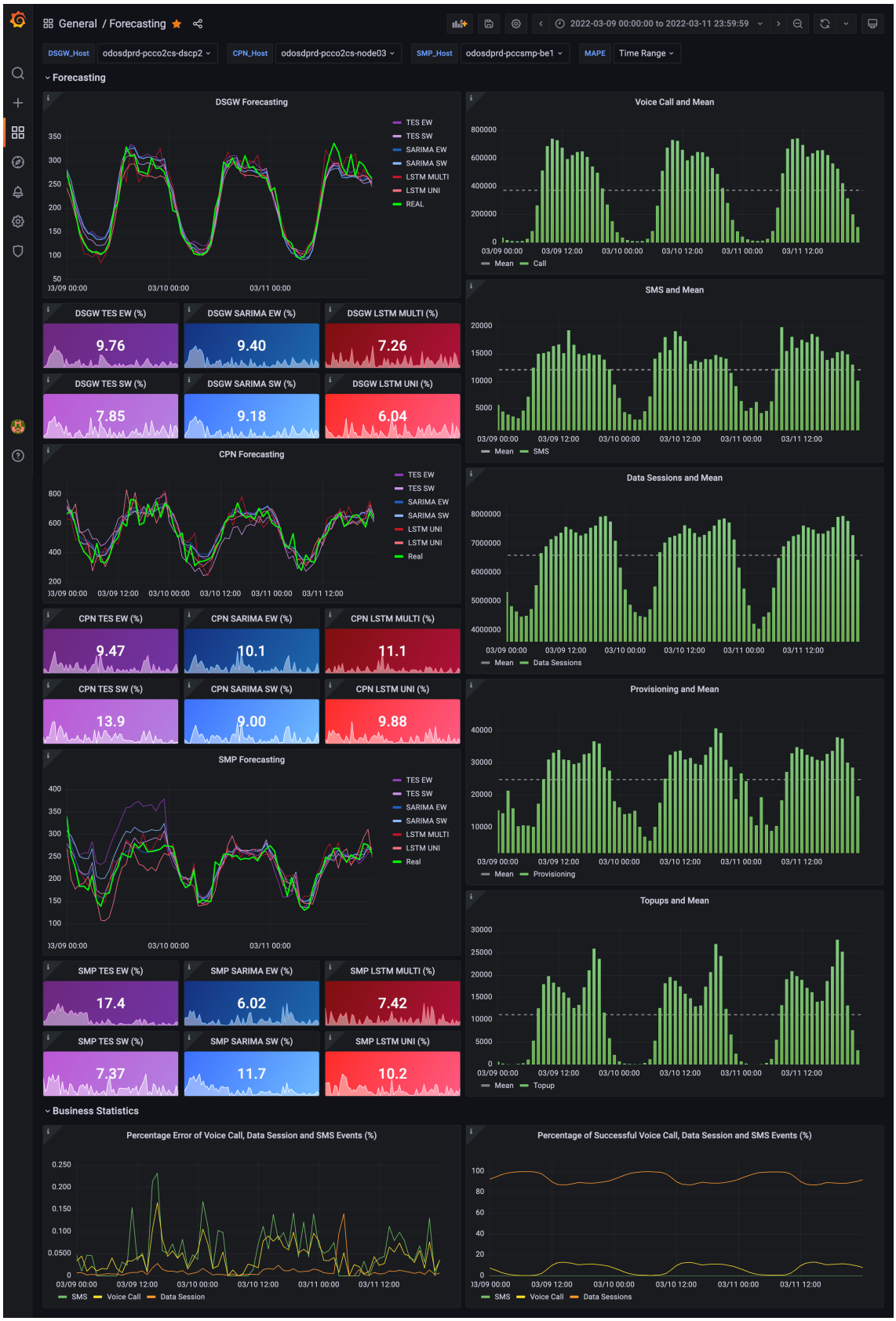


Figure 5.25: Grafana Full Dashboard for Machine 1 All Sites

mean value to guide the viewer into interpreting if the selected data behaves the same as the rest of the data during the time range.

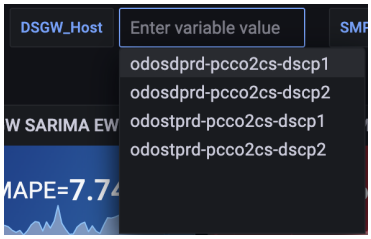


Figure 5.28: DSGW_Host Options

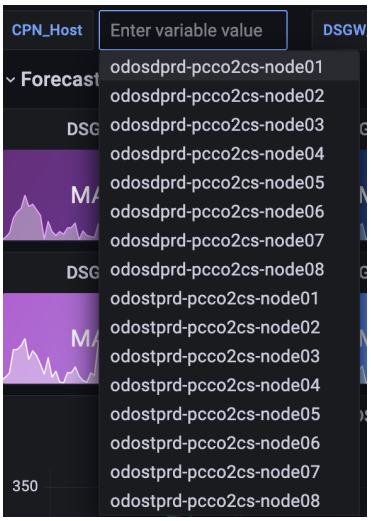


Figure 5.29: CPN_Host Options

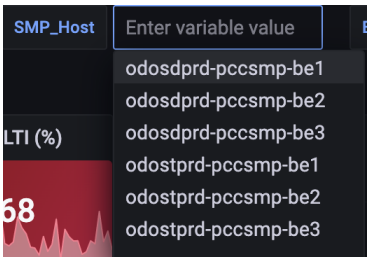


Figure 5.30: SMP_Host Options

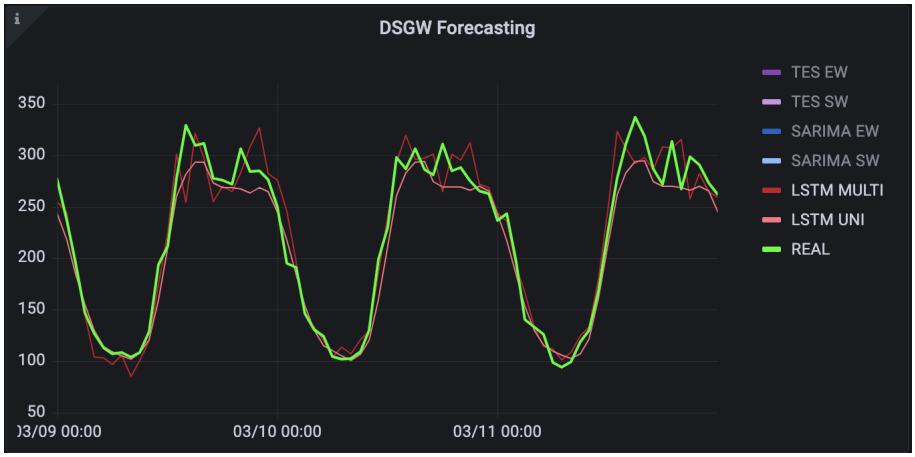


Figure 5.31: Time Line Graph Data Selection

Lastly, at the end of the dashboard, there is a set of "Business Statistics" where we can have an idea of the proportion of the type of requests. As explained in the subsubsection 3.4.2 the type of business feature with higher requests is data session, which is the feature that most influences the telecommunication system. On the right, we have a percentage error of the total request per business that triggered an error. Regarding proportion, SMS and voice calls generated more errors, but the absolute number is higher regarding data sessions because it also has the highest number of requests.

5.5 Summary

The forecasting processing and results were displayed in this chapter from the previous chapters. In order to make the forecasting, two approaches were used, the traditional time series algorithms and a machine learning adaptation.

One of the advantages of using machine learning approaches is that more features can be used in order to improve forecasting, which is called multivariate data. In order to use this kind of data, we need to choose the features with the highest correlation and the ones that create the highest impact on the CPU Load, which is the feature we are trying to predict, as can be seen on section 5.1.

In order to rank the set of features RF was used, and then to choose the features, an elbow graph was produced by training the model by adding one feature at a time and getting its error. In the end, the number of selected features was reduced to around 50%. After looking up with more attention to the features at each site, we saw that there are features in common at each site and even between sites, some of which are related to idle, user, system, and interrupt.

First LSTM was used to forecast using a neuronal structure shown in articles read before. Then, the same structures were used for multivariate and univariate data to see if adding more features decreased or not the error. It was proved that using LSTM with multivariate data improved for almost all machines at all sites, as seen in Table 5.22. If we look at machine one at CPN site LSTM using univariate data resulted in MAPE of 16.98% which is also the worst result among the rest, and using multivariate data resulted in 15.5%, which is an improvement of 1.58%.

Although using multivariate data adds more value to the resulting forecast, when adapting Machine Learning (ML) into forecasting, it loses all the perks time produces, like trend and seasonality.

The second part was forecasting using traditional time series and verifying what would result in better results. Here two approaches were used SARIMA and TES. For each one, different train sizes approaches were used, expanding window, which gives the opportunity to train with more outliers and longer patterns, as well as sliding window to decrease the time training time that increases with time.

For SARIMA, a manual process called Box-Jenkins and a grid-search approach to finding the parameters were shown. The best combination was using a manual approach to define some fixed parameters to make the grid-search process quicker. The models resulted were the simpler ones between 0 and 1 which are even more straightforward than the ones chosen manually.

SARIMA using expanding window is the approach with the most significant time to train as seen in Table 5.25. However, the forecasting results were better, 13.71%, an improvement of 3.37% compared to the worse model. Furthermore, it was found that using a sliding window approach with the biggest size, meaning one-month training data, generated lower errors. For machine one at site CPN, as well as for almost all of the machines at each site, this approach was the one that,

compared to the rest, generated lower errors, 13.24%, corresponding to a decrease MAPE of 3.74%.

Finally, another approach was used. This time, besides considering the time components like seasonality and trend, it could integrate the level of importance. Unfortunately, there is no generalization when it comes to TES parameters. Besides that, it is not as good an approach as SARIMA and is the worst approach to use most of the time. Compared to the previous approves, the expanding window generated an error of 14.2%, corresponding to a decrease of 2.78% of MAPE compared to the worst one for machine one at the CPN site, but is higher than both SARIMA approaches. The same grid-search to find the best sliding window size was implemented, with no straightforward generalization besides that at site CPN, the size 168 generated the worse results and 72 the best ones. The worst size at DSGW was 48, and there was no precise best size. Finally, SMP with size 48 generated the worse results and 168 the best. If we look at machine one at site CPN, it resulted in 16.31%, which is almost the worst result by a difference of 0.67%. Even though TES generated the worst results when looking at Table 5.25 we see that it was the one with the lowest training time.

Answering the fourth question from the chapter 1, we concluded that SARIMA is the overall best model to use, even if we compare overall error to data without outliers or only at the outliers point. This can be conferred in the Table 5.22,Table 5.23 and Table 5.24. If we use a sliding window for the present dataset, it looks like using only one month is the best approach when forecasting CPU Load. It makes complete sense because compared to the other algorithms when we use LSTM, we lose the time perks, and if we use TES, we apply importance to the most recent or the historical data. Only one month is enough because if high outliers are presented in the training data, the resulting forecasting error can get higher.

The final forecast and business data can be seen in a final dashboard in Grafana. All the features in a dashboard offer a better understanding of the system.

All Models Error for All Machine on CPN site						
Machine	TES EW	TES SW	SARIMA EW	SARIMA SW	LSTM UNI	LSTM MULTI
1	14.2	16.31	13.71	13.24	16.98	15.4
2	10.01	13.33	8.67	8.57	9.44	9.13
3	11.42	16.22	10.32	10.12	11.24	11.4
4	10.46	14.61	10.18	9.84	11.89	10.81
5	11.81	14.12	10.15	9.86	10.58	10.41
6	10.82	14.43	10.07	9.83	11.3	10.32
7	10.22	14.29	9.33	8.93	9.95	9.56
8	10.89	15.12	10.12	9.87	10.89	10.3
9	15.52	17.95	12.88	13.19	12.04	13.21
10	13.45	18.55	12.37	12.37	12.41	12.45
11	12.46	15.31	11.58	11.35	9.6	11.87
12	12.38	15.4	11.2	10.95	13.0	11.64
13	12.33	15.75	10.96	10.89	10.75	10.85
14	13.32	16.34	11.82	11.51	13.34	12.58
15	12.35	14.89	10.53	10.28	10.99	11.25
16	12.62	15.01	11.18	10.95	12.32	11.15

Table 5.22: All Models Error for All Machine (CPN): higher values colored in red and lower values colored in green each size

All Models Error for All Machine on DSGW site						
Machine	TES EW	TES SW	SARIMA EW	SARIMA SW	LSTM UNI	LSTM MULTI
1	16.45	14.13	13.02	15.3	10.94	11.19
2	18.99	16.43	11.8	11.91	8.75	9.9
3	14.84	10.99	10.46	9.92	11.23	9.32
4	13.04	10.31	10.13	10.06	9.9	9.02

Table 5.23: All Models Error for All Machine (DSGW): higher values colored in red and lower values colored in green each size

All Models Error for All Machine on SMP site						
Machine	TES EW	TES SW	SARIMA EW	SARIMA SW	LSTM UNI	LSTM MULTI
1	11.6	9.24	7.49	7.9	8.98	8.33
2	11.25	9.5	8.82	8.94	9.52	8.41
3	11.39	9.83	8.99	9.16	9.28	8.7
4	14.84	10.72	9.64	9.85	12.43	10.92
5	13.76	10.36	10.7	10.25	10.37	10.77
6	14.67	10.88	10.81	10.53	12.43	10.99

Table 5.24: All Models Error for All Machine (SMP): higher values colored in red and lower values colored in green each size

Training time in seconds for each forecasting model approach					
SARIMA EW	SARIMA SW	TES EW	TES SW	LSTM UNI	LSTM MULTI
29.86	5.61	0.57	0.23	20.12	19.45

Table 5.25: All Models Training Time in seconds

Chapter 6

Conclusion and Future Work

In this dissertation, it was defined what telecommunication means and how much it is essential nowadays. The reason is that the infrastructure that it disposes of is needed in most of the services and technology we use daily.

It presented the dataset on which this dissertation was based, composed of two data types: operational and business. The hardware information is gathered in the operational data, and statistics about the requests for each site are gathered for CPN, DSGW, and SMP. Regarding the business data, we have the number of requests for each type of message, voice call, and data session. The goal is to extract valuable information from this data and forecast the most critical feature with more inconsistencies to optimize the resources.

Briefly, it was explained where data were extracted and what kind of data we can expect to get. Then, the problem definition was introduced to understand the methodologies proposed within the dissertation's scope. The problem consists of receiving multiple tabular time series from which there is a need to understand its peaks, tendencies, their tendency to change, and how they affect the charging process.

Succinctly explained the background, most of the algorithms were discussed in the related work, why they are important, and how we can evaluate them. It was explained that the evaluation metrics are not the same depending on the output type, and therefore other strategies were exposed. For this dissertation Mean Square Error (MSE) is used to penalize the outliers, and Mean Absolute Percentage Error (MAPE) is an auxiliary metric to help with the interpretation. Based on the background concepts and definition analysis, we consider two types of forecasting using machine learning or traditional time series.

In the chapter about related work, the two forecasting types are shown by adapting Machine Learning (ML) or by using time series approaches as well as the most used metric to evaluate the forecasting results. The best performance technique when forecasting using machine learning was LSTM. For traditional time series, Triple Exponential Smoothing or Holt-Winters (TES) and Seasonal Autoregressive Integrated Moving Average (SARIMA) were used because they used time series statistics to perform long-time forecasting. Here it was concluded to use

these techniques as well as MAPE and MSE as metrics to evaluate the forecasting result. These metrics were used to obtain parameters, validate the model, and compare it at the end with the other models.

Finally, the forecasting methodology and results were described, where all the pre-processing, forecasting, and results classification were explained and shown with the help of a scheme. Different processes are used to forecast the CPU load as two different approaches. For the Long Short-Term Memory (LSTM) forecasting, there is the opportunity to take advantage of other features in order to decrease the MAPE.

When it comes to LSTM, univariate and multivariate data were used to forecast the CPU Load. In order to take advantage of other features to use in a multivariate case, there is the need to select the features that have a positive impact on the model. For that, the Random Forest (RF) is used to rank and therefore select the ones with the lowest error. This process was repeated for every machine at all sites. After looking in detail at each site's features, we saw that there are features in common at each site and even between sites, some of which are related to idle, user, system, and interrupt. When using LSTM when using a multivariate data MAPE decreased for almost all scenarios. For example, looking at machine 1 for site CPN, the error decreased from 16.98% to 15.5% compared to using multivariate data instead of univariate. Unfortunately later on it was found that the time statistics like trend and seasonality led to lower MAPE rather than more features.

When it comes to time series approaches SARIMA and TES different types of the train are used expanding and sliding window. This last approach is used to overcome the increasing training time when using expanding window. In order to classify the results, as explained in the approach, the goal is to implement a solution to forecast anomalies. When extracting the overall error, the anomalies are suppressed because there are more values within the normal range than anomalies. Three different thresholds were used to classify the anomalies and, therefore, the forecasting models with or without them. When data is classified without anomalies, it can be seen how well the model forecast an ideal scenario where the values are similar for a day. For SARIMA, it was used Box Jenkins (BJ) to fix some parameters and grid search to find out the best model for each machine at each site. Looking at machine one SARIMA using an expanding without resulted in 13.71% and using a sliding window led to a MAPE of 13.24% which is the best result for machine one at site CPN during this dissertation. It was found that using a training window of 1 month was enough to forecast the CPN Load because there were some peaks in the data that led to higher errors.

After SARIMA forecasting TES was tested because it would bring the element of importance to data, seasonality, and trend. This model led to some of the worse results for some machines meaning that data should remain with the same weight for all training data as it was done for SARIMA. Expanding the window compared to the previous approaches generated an error of 14.2%, which corresponds to an improvement of 2.78% compared to the worst one for machine one at the CPN site, but it is higher than both SARIMA approaches. The same grid-search to find the best sliding window size was implemented, looking at machine one at site CPN it resulted in 16.31% which is almost the worst results by a differ-

ence of 0.67% of the LSTM using univariate data.

In the end, the final dashboard can be seen where all the forecasting results for all machines at each site can be compared as well as two types of error, whether it is MAPE calculated for the whole dataset or whether it is for the time range selected.

Answering all the questions proposed in the chapter 1 CPU Load does not evolve the same way in time for all machines at all sites. So a different model should be trained and validated for each machine. The target feature to be forecasting is CPU Load since it presents the mode inconsistencies in tendencies and no clear outliers patterns. The days with the highest request depend on their type. Mainly, there are more daily requests than telecommunication used at night. When looking at holidays and weekends, voice calls and SMS decrease but the data session increases. To conclude, the best forecasting approach is SARIMA using a sliding window approach with a fixed training size of one month is the overall best model to use, and the worst is TES. Comparing the best model SARIMA with the worst model TES for machine one at the CPN site using both sliding window, the error decreases by 2.78%. Finally, an automated tracking visualization was created using Grafana. This tool can connect to the database and update the visualization as the database is updated as well. In this dashboard, the best models for each forecasting algorithm are visualized compared to the actual value. In addition, there is the ability to see these forecasting values alongside the business data for each machine at each site.

All the goals for this dissertation were fulfilled, starting with several approaches for forecasting the essential feature, the CPU Load, and the final dashboard with an automated update using a connection to the database. All the processes described during the dissertation are almost all automated. The human involvement will depend on merging the forecasting module to the Next Generation Intelligent Network (NGIN).

This dissertation's main difficulty was understanding the data, its values, and how to approach forecasting. Compared to the initial plan, some topics were overestimated, others underestimated. Forecasting can sometimes be miscalculated as we need to understand the primary data we are focusing on at the first step and, with time, add or even remove some features that do not affect the forecasting positively. Moreover, telecommunications is just the tip of the iceberg, all the processes underlying and infrastructure can sometimes be superseded depending on the expertise of the person or team who is doing the forecasting.

Before stepping right into the models to use or which forecasting will be used, we need to understand the time series itself. Depending on its properties, we should gather all the options, such as traditional time series algorithms or machine learning. For each one of these models, we need to study their setting, their parameters, and how they can we use it applied to the time series. The testing step was underestimated because when dealing with time series, it can take some time to train and fit the models as we do in the sliding window and for different training sizes.

Although a suitable model was found, it should be tested for a long time since

the data used for this dissertation was composed of less than a year. In future work, the models can also be tested with more complex parameters and verify that results in a better model. Furthermore, it should create a dynamic model of updating the parameters from time to time to re-calibrate the model and adapt to world changes. Besides the models tested during this dissertation, multiple other algorithms will be evaluated using this data and, of course, eventually, add more features that decrease the error.

References

[1] G. Mahalakshmi, S. Sridevi, and S. Rajaram. A survey on forecasting of time series data. In *2016 International Conference on Computing Technologies and Intelligent Data Engineering (ICCTIDE'16)*, pages 1–8, 2016. doi: 10.1109/ICCTIDE.2016.7725358.

[2] Filip Nääs Starberg and Axel Rooth. Predicting a business application's cloud server cpu utilization using the machine learning model lstm, 2021.

[3] Jianhuang Liang, Jian Cao, Jungang Wang, and Yuxia Xu. Long-term cpu load prediction. In *2011 IEEE Ninth International Conference on Dependable, Autonomic and Secure Computing*, pages 23–26, 2011. doi: 10.1109/DASC.2011.28.

[4] Martin Duggan, Karl Mason, Jim Duggan, Enda Howley, and Enda Barrett. Predicting host cpu utilization in cloud computing using recurrent neural networks. In *2017 12th International Conference for Internet Technology and Secured Transactions (ICITST)*, pages 67–72, 2017. doi: 10.23919/ICITST.2017.8356348.

[5] Karl Mason, Martin Duggan, Enda Barrett, Jim Duggan, and Enda Howley. Predicting host cpu utilization in the cloud using evolutionary neural networks. *Future Generation Computer Systems*, 86, 04 2018. doi: 10.1016/j.future.2018.03.040.

[6] Spyros Makridakis, S. Wheelwright, and Rob Hyndman. *Forecasting: Methods and Applications*, volume 35. 01 1984. doi: 10.2307/2581936.

[7] Neal Wagner and Zbigniew Michalewicz. An analysis of adaptive windowing for time series forecasting in dynamic environments: Further tests of the dyfor gp model. pages 1657–1664, 01 2008. doi: 10.1145/1389095.1389406.

[8] Indranil Bose and Radha K. Mahapatra. Business data mining — a machine learning perspective. *Information & Management*, 39(3):211–225, 2001. ISSN 0378-7206. doi: https://doi.org/10.1016/S0378-7206(01)00091-X. URL `https://www.sciencedirect.com/science/article/pii/S037872060100091X`.

[9] Junxingxu Chen, Ting Li, Yanhui Zou, Guorui Wang, Huisheng Ye, and Fengyi Lv. An ensemble feature selection method for short-term electrical load forecasting. In *2019 IEEE 3rd Conference on Energy Internet and Energy System Integration (EI2)*, pages 1429–1432, 2019. doi: 10.1109/EI247390.2019.9062042.

[10] Arie Nugroho, Ahmad Zainul Fanani, and Guruh Fajar Shidik. Evaluation of feature selection using wrapper for numeric dataset with random forest algorithm. In *2021 International Seminar on Application for Technology of Information and Communication (iSemantic)*, pages 179–183, 2021. doi: 10.1109/iSemantic52711.2021.9573249.

[11] J K Das. Business forecasting: Box-jenkins methodology. 01 2015.

[12] Sriram N Rao, G Shobha, Srinivas Prabhu, and N Deepamala. Time series forecasting methods suitable for prediction of cpu usage. In *2019 4th International Conference on Computational Systems and Information Technology for Sustainable Solution (CSITSS)*, volume 4, pages 1–5, 2019. doi: 10.1109/CSITSS47250.2019.9031015.

[13] Sabrina Belaid Boualit and Adel Mellit. Sarima-svm hybrid model for the prediction of daily global solar radiation time series. In *2016 International Renewable and Sustainable Energy Conference (IRSEC)*, pages 712–717, 2016. doi: 10.1109/IRSEC.2016.7983867.

[14] Umair F. Siddiqi, Sadiq M. Sait, and Okyay Kaynak. Genetic algorithm for the mutual information-based feature selection in univariate time series data. *IEEE Access*, 8:9597–9609, 2020. doi: 10.1109/ACCESS.2020.2964803.

[15] Hyun Kang. The prevention and handling of the missing data. *Korean journal of anesthesiology*, 64:402–6, 05 2013. doi: 10.4097/kjae.2013.64.5.402.

[16] B.G. Amidan, Thomas Ferryman, and Scott Cooley. Data outlier detection using the chebyshev theorem. pages 3814 – 3819, 04 2005. doi: 10.1109/AERO.2005.1559688.

[17] S.R. Baig and Universitat Politècnica de Catalunya. Departament d'Arquitectura de Computadors. *Data Center's Telemetry Reduction and Prediction Through Modeling Techniques*. Universitat Politècnica de Catalunya, 2020. URL `https://books.google.pt/books?id=8x2BzQEACAAJ`.

[18] Shuja-Ur-Rehman Baig, Waheed Iqbal, Josep Lluis Berral, Abdelkarim Erradi, and David Carrera. Adaptive prediction models for data center resources utilization estimation. *IEEE Transactions on Network and Service Management*, 16(4):1681–1693, 2019. doi: 10.1109/TNSM.2019.2932840.

[19] Ali Asghar Rahmanian, Mostafa Ghobaei-Arani, and Sajjad Tofighy. A learning automata-based ensemble resource usage prediction algorithm for cloud computing environment. *Future Generation Computer Systems*, 79:54–71, 2018. ISSN 0167-739X. doi: https://doi.org/10.1016/j.future.2017.09.049. URL `https://www.sciencedirect.com/science/article/pii/S0167739X17309378`.

[20] Lingyun Yang, I. Foster, and J.M. Schopf. Homeostatic and tendency-based cpu load predictions. In *Proceedings International Parallel and Distributed Processing Symposium*, pages 9 pp.–, 2003. doi: 10.1109/IPDPS.2003.1213129.

[21] Behrouz Banitalebi, Md. Erfanul Hoque, Srimantoorao S. Appadoo, and Aerambamoorthy Thavaneswaran. Regularized probabilistic forecasting of electricity wholesale price and demand. In *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 28–35, 2020. doi: 10.1109/SSCI47803.2020.9308333.

[22] A.S. Karaman and T. Altiok. An experimental study on forecasting using tes processes. In *Proceedings of the 2004 Winter Simulation Conference, 2004.*, volume 1, page 442, 2004. doi: 10.1109/WSC.2004.1371346.

[23] Neda Akrami, Koorush Ziarati, and Soumyabrata Dev. Predicting temperature from ground-based synoptic data in shiraz city, iran. In *2021 Photonics & Electromagnetics Research Symposium (PIERS)*, pages 2141–2146, 2021. doi: 10.1109/PIERS53385.2021.9695127.

[24] Ebru Idman, Emrah Idman, and Osman Yildirim. Estimating solar power plant data using time series analysis methods. In *2020 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA)*, pages 1–6, 2020. doi: 10.1109/HORA49412.2020.9152839.

[25] Ali Lahouar, Amal Mejri, and Jaleleddine Ben Hadj Slama. Importance based selection method for day-ahead photovoltaic power forecast using random forests. In *2017 International Conference on Green Energy Conversion Systems (GECS)*, pages 1–7, 2017. doi: 10.1109/GECS.2017.8066171.

[26] Ji Sung Lee, Hyeon Sung Cho, Kyo Il Chung, and Ji Sang Park. Feature selection for stock forecasting using multivariate convolution neural network. In *2020 International Conference on Information and Communication Technology Convergence (ICTC)*, pages 1270–1272, 2020. doi: 10.1109/ICTC49870.2020.9289492.

[27] Shile Chen and Changjun Zhou. Stock prediction based on genetic algorithm feature selection and long short-term memory neural network. *IEEE Access*, 9:9066–9072, 2021. doi: 10.1109/ACCESS.2020.3047109.

[28] Yu-Wei Chang and Chih-Yung Tsai. Apply deep learning neural network to forecast number of tourists. In *2017 31st International Conference on Advanced Information Networking and Applications Workshops (WAINA)*, pages 259–264, 2017. doi: 10.1109/WAINA.2017.125.

[29] Rico S. Santos and Davood Pour Yousefian Barfeh. Time series forecasting for issuance of alien employment permits by nationality in the philippines using holt-winters method. In *2019 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE)*, pages 240–245, 2019. doi: 10.1109/ICCIKE47802.2019.9004243.

[30] Marina Soley-Bori. Dealing with missing data: Key assumptions and methods for applied analysis.

[31] Corvin Idler. *Pattern Recognition and Machine Learning Techniques for Algorithmic Trading*. PhD thesis, 07 2014.

[32] Hari Luitel and Gerry Mahar. Testing for unit roots in autoregressive-moving average models of unknown order: Critical comments*. *SSRN Electronic Journal*, 04 2017. doi: 10.2139/ssrn.2882101.

Appendices

Appendix A

Dataset Full Description

Feature Selection	Description
from_date	starting range time with format yyyy:mm:dd:hh
to_date	ending range time with format yyyy:mm:dd:hh
host	categorical variable with name of the machine from site CPN, DSGW and SMP
kpi	key performance indicator of the data of CPN, DSGW and SMP, kpi="CPU"
metric	categorical variable representing a CPU metric: idle: percentage of CPU with no runnable process
	interrupt: percentage of CPU dedicated to hardware interrupts
	iowait: percentage of CPU used for completion of I/O activities
	nice: percentage of CPU spent on low priority processes
	steal: percentage of CPU when it had something runnable, but the XEN hypervisor chose to run something else instead
	system: percentage of CPU running kernel processes
	user: percentage of CPU running non kernel processes
min	minimum percentage of CPU usage of the specific metric in a minute between from_date and to_date
avg	average percentage of CPU usage of the specific metric in a minute between from_date and to_date
max	maximum percentage of CPU usage of the specific metric in a minute between from_date and to_date

Table A.1: Description of the CPN, DSGW and SMP CPU Dataset: Percentage of CPU usage by computations of the application in a minute for each hour of a day

Feature Selection	Description
from_date	starting range time with format yyyy:mm:dd:hh
to_date	ending range time with format yyyy:mm:dd:hh
host	categorical variable with name of the machine from site CPN, DSGW and SMP
kpi	key performance indicator of the data of CPN, DSGW and SMP, kpi="CPU_LOAD"
metric	categorical variable representing the number of waiting processes in 15min, 5min and 1min [status[15], status[5], status[1]]
min	minimum percentage of number of waiting processes for a specific amount of minutes between from_date and to_date
avg	average percentage of number of waiting processes for a specific amount of minutes between from_date and to_date
max	maximum percentage of number of waiting processes for a specific amount of minutes between from_date and to_date

Table A.2: Description of the CPN, DSGW and SMP CPU_LOAD Dataset: Number of waiting processes for 1,5 and 15 minutes each hour of the day

Feature Selection	Description
from_date	starting range time with format yyyy:mm:dd:hh
to_date	ending range time with format yyyy:mm:dd:hh
host	categorical variable with name of the machine from site CPN, DSGW and SMP
kpi	key performance indicator of the data of CPN, DSGW and SMP, kpi="logging"
metric	categorical variable representing the different types of logs generated by application CPN, DSGW and SMP: results[DEBUG]:compiled out of Release builds
	results[ERROR]:problem that should be investigated
	results[INFO]:important information that should be logged
	results[TRACE]:provide context to understand the steps leading up to errors and warnings
	results[WARN]:this MIGHT be problem, or might not
	total:total number of logs
min	minimum number of logs generated by application CPN, DSGW and SMP for a specific type of results in a minute between from_date and to_date
avg	average number of logs generated by application CPN, DSGW and SMP for a specific type of results in a minute between from_date and to_date
max	maximum number of logs generated by application CPN, DSGW and SMP for a specific type of results in a minute between from_date and to_date
sum	sum number of logs generated by application CPN, DSGW and SMP for a specific type of results in a minute between from_date and to_date

Table A.3: Description of the CPN, DSGW and SMP LOGGING Dataset: Number of logs generated for each type of result

Feature Selection	Description
from_date	starting range time with format yyyy:mm:dd:hh
to_date	ending range time with format yyyy:mm:dd:hh
kpi	key performance indicator of the data of CPN, kpi=["OCS_KERNEL_Result_CHRG_Diameter", "QRP_PN_INPUTREQS_RESULT"]
metric	categorical variable representing the different types of logs generated by application CPN: rt_err:event type from kpi = "QRP_PN_INPUTREQS_RESULT" , requests that resulted in error
	rt_suc:event type from kpi = "QRP_PN_INPUTREQS_RESULT", successful requests results
	rt_others:event type from kpi = "QRP_PN_INPUTREQS_RESULT", requests that resulted in other types of logs
	total:total events produced by from kpi = "QRP_PN_INPUTREQS_RESULT"
	results[final]:event type from kpi = "OCS_KERNEL_Result_CHRG_Diameter"
	results[intermediate]:event type from kpi = "OCS_KERNEL_Result_CHRG_Diameter"
	results[unique]:event type from kpi = "OCS_KERNEL_Result_CHRG_Diameter"
	results[unknown]:event type from kpi = "OCS_KERNEL_Result_CHRG_Diameter"
	results[OTHERS]:event type from kpi = "OCS_KERNEL_Result_CHRG_Diameter"
min	minimum number of logs generated by application CPN for a specific type of results in a minute between from_date and to_date
avg	average number of logs generated by application CPN for a specific type of results in a minute between from_date and to_date
max	maximum number of logs generated by application CPN for a specific type of results in a minute between from_date and to_date
sum	sum number of logs generated by application CPN for a specific type of results in a minute between from_date and to_date

Table A.4: Description of the CPN TOR_CSR Dataset: Number of logs generated for each type of result

Feature Selection	Description
from_date	starting range time with format yyyy:mm:dd:hh
to_date	ending range time with format yyyy:mm:dd:hh
kpi	key performance indicator of the data of DSGW for data session, voice call and sms, kpi=["Result Codes Gy", "Diameter Result Codes Ro APC","Diameter Result Codes Ro SMS"]
metric	categorical variable representing the different types of logs generated by application DSGW:
	rt_err:event type for all KPIs , requests that resulted in error
	rt_suc:event type for all KPIs, successful requests results
	rt_others:event type for all KPIs, other requests results
	total:total events type for all KPIs
min	minimum number of logs generated by application CPN for a specific type of results in a minute between from_date and to_date
avg	average number of logs generated by application CPN for a specific type of results in a minute between from_date and to_date
max	maximum number of logs generated by application CPN for a specific type of results in a minute between from_date and to_date
sum	sum number of logs generated by application CPN for a specific type of results in a minute between from_date and to_date

Table A.5: Description of the CPN DCR Dataset: Number of logs generated for each type of result

Feature Selection	Description
from_date	starting range time with format yyyy:mm:dd:hh
to_date	ending range time with format yyyy:mm:dd:hh
kpi	key performance indicator of the data of DSGW for data session, voice call and sms, kpi=["Result Codes Gy", "Diameter Result Codes Ro APC","Diameter Result Codes Ro SMS"]
min	minimum number of voice call, sms and data session requests in a minute between from_date and to_date
avg	average number of voice call, sms and data session requests in a minute between from_date and to_date
max	maximum number of voice call, sms and data session requests in a minute between from_date and to_date
sum	sum number of voice call, sms and data session requests in a minute between from_date and to_date

Table A.6: Description of the CPN DRR Dataset: Number of logs generated for each type of result

Feature Selection	Description
from_date	starting range time with format yyyy:mm:dd:hh
to_date	ending range time with format yyyy:mm:dd:hh
kpi	key performance indicator of the data of SMP for data session, voice call and sms, kpi="wtr_result_ocs"
metric	categorical variable representing the different types of logs generated by application SMP: rt_err:event type for all KPIs , requests that resulted in error
	rt_suc:event type for all KPIs, successful requests results
	rt_others:event type for all KPIs, other requests results
	total:total events type for all KPIs
min	minimum number of voice call, sms and data session requests in a minute between from_date and to_date
avg	average number of voice call, sms and data session requests in a minute between from_date and to_date
max	maximum number of voice call, sms and data session requests in a minute between from_date and to_date
sum	sum number of voice call, sms and data session requests in a minute between from_date and to_date

Table A.7: Description of the SMP RCOCS Dataset: Number of request for each type of result