

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Machine Learning Algorithm for Assessing Partial Shading Losses based on Inverter Data

Tiago Francisco Rodrigues Pires



Mestrado em Engenharia Eletrotécnica e de Computadores

Supervisor: Armando Luís Sousa Araújo

July 26, 2024

Resumo

O sombreamento parcial em painéis solares leva a uma redução na produção e ao desgaste acelerado dos mesmos, colocando desafios significativos à eficiência do sistema fotovoltaico. Assim, esta dissertação documenta o desenvolvimento e avaliação de dois modelos distintos de machine learning para identificar e avaliar o impacto do sombreamento parcial numa central fotovoltaica real no sul de Portugal. A produção de potência esperada, calculada a partir da irradiância medida e dos parâmetros dos painéis solares, serve como referência contra a produção medida.

O primeiro modelo utiliza a técnica de clustering *K-means* para analisar as diferenças entre a produção esperada e a medida, agrupando dados com base nestas discrepâncias. O segundo modelo utiliza os resultados do modelo *K-means* como *labels* para uma rede neuronal *Long Short-Term Memory* (LSTM), que classifica os períodos de sombreamento parcial. São apresentados dados experimentais de ambos os modelos, com o modelo *K-means* a alcançar uma aproximação mais próxima dos valores de referência, embora os modelos LSTM tenham demonstrado flexibilidade e escalabilidade sem necessitar de conhecimento prévio do conjunto de dados por parte do utilizador final.

Os resultados destacam o potencial destes modelos como ferramentas de diagnóstico para a manutenção e otimização de centrais fotovoltaicas, contribuindo para uma produção de energia mais eficiente e uma gestão de recursos mais eficaz. Alguns do trabalho a realizar no futuro será a otimização de hiperparâmetros, a exploração de novas arquiteturas de redes neurais e a integração de funcionalidades adicionais para estimar o impacto económico do sombreamento parcial. Estes avanços estão alinhados com os Objetivos de Desenvolvimento Sustentável das Nações Unidas, promovendo soluções energéticas mais limpas e eficientes.

Keywords: Sombreamento Parcial, Painéis Solares, Sistemas Fotovoltaicos, Machine Learning, *K-Means Clustering*, Redes Neurais LSTM, Energias Renováveis.

Abstract

Partial shading in solar panels leads to reduced production and accelerated equipment ageing, posing significant challenges to the photovoltaic system's efficiency. This dissertation documents the development and evaluation of two distinct machine-learning models to identify and assess the impact of partial shading in a real-world photovoltaic power plant in southern Portugal. The expected power output, calculated from measured irradiance and solar panel parameters, is a benchmark against the actual measured power.

The first model employs K-means clustering to analyse the differences between expected and measured power, grouping data based on these deviations. The second model leverages the outputs of the K-means model as labels for a Long Short-Term Memory (LSTM) neural network, which classifies periods of partial shading. Experimental data from both models are presented, with the K-means model achieving a closer approximation to the reference values. However, the LSTM models demonstrated flexibility and scalability without requiring prior dataset knowledge from the end user.

The findings highlight the potential of these models as diagnostic tools for photovoltaic power plant maintenance and optimisation, contributing to more efficient energy production and resource management. Future work includes hyperparameter optimisation, exploring new neural network architectures, and integrating additional features to estimate the economic impact of partial shading. These advancements align with the United Nations' Sustainable Development Goals by promoting cleaner, more efficient energy solutions.

Keywords: Partial Shading, Solar Panels, Photovoltaic Systems, Machine Learning, K-Means Clustering, LSTM Neural Networks, Renewable Energy.

United Nations' Sustainable Development Goals (SDGs)

From the 25th to the 27th of September 2015, the Heads of State and Government and High Representatives met at the United Nations Head Quarters to celebrate the organisation's 70th anniversary. Here, the participating entities adopted the 2030 Agenda for Sustainable Development. This Agreement is comprised of 17 major goals, which are in turn divided into 169 targets. This universal agenda balances the three dimensions of sustainable development: social, economic and environmental.

This work, integrated into the field of renewable energies, aims to help make solar energy production more efficient. Partial shading hastens equipment decay, which in turn will require substitutions at a faster rate. This wasteful management of resources and energetic losses equates to economic losses for the interested parties and a larger carbon footprint, waste generation and energetic inefficiency.

By highlighting the magnitude of these losses, we aim to promote environmentally mindful practices amongst the concerned decision-makers. These practices, namely prevention, will help reduce negative impacts on resource management and help photovoltaic power plant owners save on equipment and energy losses. EPC companies save on PR-associated penalties, proving beneficial for all parties involved.

SDG	Target	How the target is met
7	7.2	Partial shading is directly responsible for the loss of clean, renewable energy that could be used instead of energy from other sources. Preventing these losses means more clean energy is injected into the electrical grid.
	7.3	Systems with fewer losses are more efficient. Prevention is the first step towards ensuring that the system is not wasteful.
12	12.2	Partial shading causes the degradation of solar cells in PV panels due to the dissipation of energy in the form of heat, which will shorten its efficiency and lifespan. Proper spacing of panels will ensure they do not require faster and, therefore, wasteful substitutions.
	12.5	Slower solar panel degradation prevents waste generation through early and avoidable equipment disposal.

Agradecimentos

Quero começar esta secção de agradecimentos por prestar o meu mais sincero ao Professor Doutor Armando Araújo, que orientou esta minha empreitada mostrando sempre disponibilidade para me auxiliar. Uma grande palavra de agradecimento também ao meu supervisor, o Engenheiro Pedro Martins, meu supervisor na CJR Renewables, empresa à qual também estendo a minha gratidão.

Estou convicto que o estudante, no pleno significado da palavra, está em vias de extinção. O aluno da faculdade busca um diploma, sem procurar dar relevo e substância à sua vida académica. Há por isso pessoas a quem agradeço terem-me encaminhado rumo a um destino que não esse.

Agradeço aos meus grandes amigos Chico, Viana, Morgado e Eduardo.

Aos: Papelão, Xauxixas, SeiSer, Lonely Kano, Azamat, Mozzarella, Acliclas, Eusébios, Min-ion Bob, Era e Machachuxas, ao meu ano, a Electro, foi uma honra ser praxista e estudante convosco. Acima de tudo, obrigado por toda a amizade que me deram e por tornarem as horas difíceis mais suportáveis.

Aos: Jota, Neca, Vasco, Juvenal, Daniel, Megazord, Carlos, Leo, Mamão, Marega, Bem Mau, Regador, Regatrês e Homem Queque, ao mui nobre Grupo de Fados da Faculdade de Engenharia da Universidade do Porto, obrigado por me acolherem. Obrigado por toda a irmandade, companheirismo e momentos vividos com estes amigos que se juntam para tocar um fadinho.

Agradeço profundamente a todos os meus amigos praxistas, fadistas e praxistas fadistas por me ensinarem o que é ser estudante e me darem uma nova família.

Não posso deixar de agradecer aos meus amigos de sempre, Alex, Kika, Barroso, JP, Bruna, Sofia, Lapa e Edu, obrigado por caminharem comigo nesta jornada. Obrigado também ao meu professor e amigo Chico D'Houtel, por quem guardo a maior estima e respeito e que me ensinou muito mais do que música.

À Faísca, obrigado pelo que me dás todos os dias, pelo apoio e força incondicionais. És o melhor que a faculdade me deu. És a minha casa longe de casa.

Aos meus pais, Jorge e Júlia, não sei como posso possivelmente agradecer toda uma vida de apoio, de carinho, de ensinamentos, de sacrifícios feitos para me verem chegar aonde estou hoje, porque sem vocês, não seria possível. À minha irmã Mariana, és dos maiores orgulhos da minha vida, obrigado por me fazeres sempre sentir que sou um dos da tua. Aos meus tios, Miguel e Cláudia, e à minha prima Leonor, obrigado por estarem sempre lá e completarem o que é para mim o conceito de família. À minha prima Sara agradeço todas as conversas e partilha, que tanto me ajudaram. À minha avó Augusta, que sei que tanto queria ter podido ver este momento, agradeço a pessoa que fizeste de mim.

Tiago Pires

*“Dude, sucking at somethin’
is the first step towards being sorta good at something ”*

Jake, the Dog

Contents

1	Introduction	1
1.1	CJR Renewables - Company Presentation	1
1.2	Context	1
1.3	Motivation	2
1.4	Objective	3
1.5	Document Structure	3
2	Background and Fundamental Aspects	4
2.1	Renewable Energies	4
2.2	Photovoltaic Power Plant Overview	5
2.2.1	Photovoltaic Modules	5
2.2.2	Inverters	6
2.3	Partial Shading Related Failures and Losses	7
2.3.1	Hot Spot Formation	7
2.3.2	Maximum Power Point Tracking Difficulties	8
2.4	Maintenance	8
3	Related Work	10
3.1	Knowledge Based Approaches	10
3.2	Machine Learning Based Approaches	11
3.2.1	Artificial Neural Networks (ANN)	12
3.2.2	Support Vector Machine (SVM)	14
3.2.3	Decision Trees (DT)	14
3.2.4	Auto-Encoder (AE)	14
3.2.5	Convolutional Neural Networks (CNN)	15
3.2.6	Research Findings in Fault Detection and Model Comparison	15
4	Methodologies and Implementations	17
4.1	Methodologies - K-Means	19
4.2	Methodologies - LSTM	23
4.2.1	Data Preprocessing	24
4.2.2	Neural Network Architecture	28
5	Results and Conclusions	32
5.1	Validation and Results	32
5.1.1	K-Means-based Models Results	32
5.1.2	LSTM-based Models Results	35
5.1.3	Model Results Comparison	36

5.2	Conclusions and Future Work	37
	References	38
	A	42
A.1	K-Means Model Runs	42
A.2	LSTM Models	61

List of Figures

1.1	PV module shaded by other modules	2
2.1	PV module equivalent circuit.	6
2.2	Characteristic I-V curve of a cell.	7
2.3	Hotspot creation	8
4.1	Measured vs Expected Power for string 2 on days 10 (TOP) and 23 (BOTTOM) in January	18
4.2	Difference between Expected and Measured Power: Day 10, string 2	19
4.3	Standard Error KMeans Clustering: string 7	20
4.4	Difference between expected and measured power in January for string 15 (TOP: 3D Perspective View of Surface Plot, BOTTOM: Top-down View of 3D Surface Plot)	21
4.5	K-means clustering applied to the loss profile of string 15 in January (TOP: 3D Perspective View of Surface Plot, BOTTOM: Top-down View of 3D Surface Plot)	22
4.6	LSTM classifier applied to the loss profile of string 15 in January, before data standardisation, showcasing false positives (TOP: 3D Perspective View of Surface Plot, BOTTOM: Top-down View of 3D Surface Plot)	25
4.7	LSTM classifier applied to the loss profile of string 15 in January, before data standardisation, showcasing poor probability attribution. (TOP: 3D Perspective View of Surface Plot, BOTTOM: Top-down View of 3D Surface Plot)	26
4.8	LSTM classifier applied to the loss profile of string 15 in January before balancing the training data set. (TOP: 3D Perspective View of Surface Plot, BOTTOM: Top-down View of 3D Surface Plot)	27
4.9	TOP: Mapped labels before training data set balancing. BOTTOM: Mapped labels after training data set balancing.	28
4.10	Final Training Loss Curve	29
4.11	Training Loss Curve with Batch Size 16	30
4.12	LSTM classifier applied to the loss profile of string 15 in January. (TOP: 3D Perspective View of Surface Plot, BOTTOM: Top-down View of 3D Surface Plot)	31
5.1	Runs 3 (TOP) and 9 (BOTTOM) of K-means clustering and Hungarian algorithm applied to the loss profile of string 15 in January.	33
5.2	Cluster 1 distribution over time, by model	34
5.3	Probability distribution, by model	36
A.1	Run 1	42
A.2	Run 2	43
A.3	Run 3	44

A.4 Run 4	45
A.5 Run 5	46
A.6 Run 6	47
A.7 Run 7	48
A.8 Run 8	49
A.9 Run 9	50
A.10 Run 10	51
A.11 Run 11	52
A.12 Run 12	53
A.13 Run 13	54
A.14 Run 14	55
A.15 Run 15	56
A.16 Run 16	57
A.17 Run 17	58
A.18 Run 18	59
A.19 Run 19	60
A.20 Run 20	61
A.21 Model 1	62
A.22 Model 2	63
A.23 Model 3	64
A.24 Model 4	65
A.25 Model 5	66
A.26 Model 6	67
A.27 Model 7	68
A.28 Model 8	69
A.29 Model 9	70
A.30 Model 10	71
A.31 Model 11	72
A.32 Model 12	73
A.33 Model 13	74
A.34 Model 14	75
A.35 Model 15	76
A.36 Model 16	77
A.37 Model 17	78
A.38 Model 18	79
A.39 Model 19	80
A.40 Model 20	81

List of Tables

2.1	Trajectory and Portugal's contribution to the European Union's 2030 binding goal.	4
2.2	Perspective for the evolution of the installed capacity for electricity per source in Portugal until 2030.	5
4.1	Total power and timestamp count per cluster for string 15 in January.	22
4.2	Total power per probability of Class 0 interval for string 15 in January	27
4.3	Total power per probability of Class 0 interval for string 15 in January	31
5.1	Results for the studied models. False positives are only counted before 15:00 . .	37
A.1	Total power and timestamp count per cluster for string 15 in January - Run 1. . .	42
A.2	Total power and timestamp count per cluster for string 15 in January - Run 2. . .	43
A.3	Total power and timestamp count per cluster for string 15 in January - Run 3. . .	44
A.4	Total power and timestamp count per cluster for string 15 in January - Run 4. . .	45
A.5	Total power and timestamp count per cluster for string 15 in January - Run 5. . .	46
A.6	Total power and timestamp count per cluster for string 15 in January - Run 6. . .	47
A.7	Total power and timestamp count per cluster for string 15 in January - Run 7. . .	48
A.8	Total power and timestamp count per cluster for string 15 in January - Run 8. . .	49
A.9	Total power and timestamp count per cluster for string 15 in January - Run 9. . .	50
A.10	Total power and timestamp count per cluster for string 15 in January - Run 10. . .	51
A.11	Total power and timestamp count per cluster for string 15 in January - Run 11. . .	52
A.12	Total power and timestamp count per cluster for string 15 in January - Run 12. . .	53
A.13	Total power and timestamp count per cluster for string 15 in January - Run 13. . .	54
A.14	Total power and timestamp count per cluster for string 15 in January - Run 14. . .	55
A.15	Total power and timestamp count per cluster for string 15 in January - Run 15. . .	56
A.16	Total power and timestamp count per cluster for string 15 in January - Run 16. . .	57
A.17	Total power and timestamp count per cluster for string 15 in January - Run 17. . .	58
A.18	Total power and timestamp count per cluster for string 15 in January - Run 18. . .	59
A.19	Total power and timestamp count per cluster for string 15 in January - Run 19. . .	60
A.20	Total power and timestamp count per cluster for string 15 in January - Run 20. . .	61
A.21	Total power per probability of Class 0 interval for string 15 in January - Model 1	62
A.22	Total power per probability of Class 0 interval for string 15 in January - Model 2	63
A.23	Total power per probability of Class 0 interval for string 15 in January - Model 3	64
A.24	Total power per probability of Class 0 interval for string 15 in January - Model 4	65
A.25	Total power per probability of Class 0 interval for string 15 in January - Model 5	66
A.26	Total power per probability of Class 0 interval for string 15 in January - Model 6	67
A.27	Total power per probability of Class 0 interval for string 15 in January - Model 7	68
A.28	Total power per probability of Class 0 interval for string 15 in January - Model 8	69
A.29	Total power per probability of Class 0 interval for string 15 in January - Model 9	70

A.30 Total power per probability of Class 0 interval for string 15 in January - Model 10	71
A.31 Total power per probability of Class 0 interval for string 15 in January - Model 11	72
A.32 Total power per probability of Class 0 interval for string 15 in January - Model 12	73
A.33 Total power per probability of Class 0 interval for string 15 in January - Model 13	74
A.34 Total power per probability of Class 0 interval for string 15 in January - Model 14	75
A.35 Total power per probability of Class 0 interval for string 15 in January - Model 15	76
A.36 Total power per probability of Class 0 interval for string 15 in January - Model 16	77
A.37 Total power per probability of Class 0 interval for string 15 in January - Model 17	78
A.38 Total power per probability of Class 0 interval for string 15 in January - Model 18	79
A.39 Total power per probability of Class 0 interval for string 15 in January - Model 19	80
A.40 Total power per probability of Class 0 interval for string 15 in January - Model 20	81

Abbreviations and symbology

1-SVM	one-class Support Vector Machine
°	Degree (Celsius)
AC	Alternating Current
AE	Auto-Encoder
AE-LSTM	AutoEncoder Long Short-Term Memory
ANN	Artificial Neural Network
B	Boron
BPTT	Backpropagation Through Time
CEC	Constant Error Carousel
CJR Renewables	CJR Renewables
CNN	Convolutional Neural Network
COP21	21st Conference of the Parties
DC	Direct Current
DC/AC	Direct Current/Alternating Current
DL	Deep Learning
DT	Decision Tree
EPC	Engineering, Procurement, and Construction
EU	European Union
FNN	Feedforward Neural Network
FL	Fuzzy Logic
GW	Gigawatt
IFR	Infrared
IMPP	Current at Maximum Power Point
I-V	Current-Voltage
ISC	Short Circuit Current
IS	Isolation Forest
kNN	k-nearest Neighbour
K-Means	(No expansion needed as it is a standard term for the clustering algorithm)
kWh/kW	Kilowatt-hour per kilowatt
kWh/m ²	Kilowatt-hour per square meter
LOF	Local Outlier Factor
LSTM	Long Short-Term Memory
ML	Machine Learning
MLP ANN	Multi-Layer Perceptron Artificial Neural Network
MPPT	Maximum Power Point Tracker
MW	Megawatt
N/A	Not Applicable
NN	Neural Network

n	Negative (charge)
O&M	Operations and Maintenance
OC	Open Circuit
PdM	Predictive Maintenance
P	Phosphorus
PM	Preventive Maintenance
PNEC 2030	Plano Nacional Energia e Clima (National Plan for Energy and Climate)
p	Positive (charge)
PR	Performance Ratio
PS	Partial Shading
PV	Photovoltaic
RF	Random Forest
RNN	Recurrent Neural Network
RM	Reactive Maintenance
SC	Short Circuit
SGD	Stochastic Gradient Descent
Si	Silicon
SVM	Support Vector Machine
UAV	Unmanned Aerial Vehicle
UN	United Nations
VMPP	Voltage at Maximum Power Point
VOC	Open Circuit Voltage
Y_F	Final Yield
Y_R	Reference Yield

Chapter 1

Introduction

This dissertation was proposed by and developed with CJR Renewables, Operation & Maintenance Department, as part of the Master in Electrical and Computer Engineering in the Faculty of Engineering of the University of Porto.

1.1 CJR Renewables - Company Presentation

Founded in 1970, the CJR Group is a multinational group and is headquartered in Guimarães, Portugal. The group's core activity was construction machinery rental until 1993 when it ventured into civil construction. In 2002, the CJR Group expanded its scope to renewable energies with the founding of CJR Renewables. The company offers EPC solutions, which stands for Engineering, Procurement and Construction services, in wind and solar energy projects and also services in the electric grid, such as transformer replacements and substation design and construction. CJR Renewables now operates in 25 countries over 3 continents and counts with 12180 MW implemented worldwide. The company has its offices in Leça do Balio, Matosinhos.

CJR Renewables is committed to the global response to the threat of climate change and enhancing sustainable development, investing in sustainable environmental practices while achieving economic performance.

1.2 Context

The urgency for clean and renewable energy has garnered attention due to the climate crisis and economic interest in solar energy, drawing focus from the scientific community in recent decades.

According to [1], several factors, such as dust accumulation or cracked panels, can lead to failures in PV modules, yielding a lower revenue due to loss of productivity. The three performance indicators for a PV system the authors presented were final yield (Y_F), reference yield (Y_R) and performance ratio (PR). The PR is the ratio between Y_F and Y_R .

Over the past decades, monitoring tools have been engineered to ensure these indicators are up to standard and intervene in the instances they are not. These tools are designed to be cheap and automated and can help to avert the average 20 % energy losses that PV systems register.

Given the large volumes of data that PV systems can generate, several machine learning-based algorithms can be found in literature [2, 3, 4, 5]. These automated applications are desirable for operators to aid in handling big data and detect anomalies in power plants that can lead to a worse PR or damage to the equipment.

1.3 Motivation

The case studied for this dissertation used the production data for a power plant built by CJR Renewables in Portugal. In the reported case, a loss of 20 % of the energy produced in a day by the shaded string was registered. This was caused due to the partial shading of the PV modules, as seen in figure 1.1.

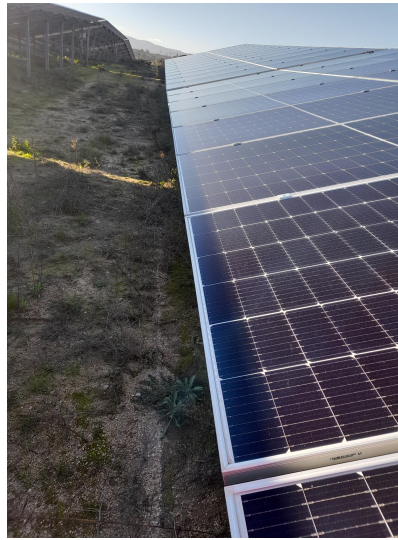


Figure 1.1: PV module shaded by other modules

The modules in front of the shaded ones cause this 10 cm shadow. The shadow lasts 2 hours, from 15:30 to 17:30. In a 31-day month, in a 30 MW power plant, that represents the loss of 300 MWh. Such losses will logically lead to a lower PR.

According to [6], when choosing the array row pitch, or distance between rows of modules, a tradeoff must be considered between the produced energy per land area (kWh/m^2) and the production per rated kilowatt (kWh/kW). Increasing the produced energy per land area might lower the production per rated kilowatt due to the loss of energy for partial shading. Reducing the production per rated kilowatt will lower the PR.

When clients commission a photovoltaic power plant, they usually demand more energy production per land area in the design phase, overlooking production per rated kilowatt. It is also

common for clients and EPC companies to have established PR values in their maintenance contracts. The EPC company may face monetary penalties when power plants perform below those PR values. However, some circumstances can exempt the EPC company from such penalties.

The problem with partial shading losses is that they are hard to explain to clients, and the issue arises due to client demands during the PV field's conception phase. When clients face significant losses and a lower PR, they demand explanations. The only way to explain shading losses is to provide an on-site picture, such as figure 1.1, and present estimates of the loss values. These explanations are often not well received and are met with scepticism.

Another source of additional expenses is maintenance routines issued to search for correct these losses. Given shading losses exist by design, these routines exist only per obligation and are, therefore, wasteful of resources and capital.

1.4 Objective

This dissertation aims to develop an algorithm to identify periods during production where partial shading is registered and assess the magnitude of the losses. It must be able to calculate the plant's expected production from the measured irradiance. The system should be able to read data from the plant's weather station (irradiance) and inverters (voltage and current) and detect anomalies in production. Once these anomalies have been identified, they must be labelled as resulting from partial shading losses.

Proper methodologies for testing the algorithm's performance must be implemented to guarantee the quality of the results, given CJR's high-quality standards.

1.5 Document Structure

This dissertation is divided into five chapters.

Chapter 1, the present chapter, aims to identify the engineering field under which this work was developed and briefly describe the problem this dissertation will address.

Chapter 2 provides the reader with the necessary knowledge and background and discusses theoretical concepts needed to understand better the work that will be conducted with this dissertation.

Chapter 3 contains a literature review conducted under the scope of the dissertation, showing recent work done in the field and serving as a guideline for this project.

Chapter 4 details the chosen algorithms, their implementation steps, and some challenges met during the experimental phase.

Chapter 5 presents the results, the conclusions that were taken from this work, and some future work-related insights.

Chapter 2

Background and Fundamental Aspects

2.1 Renewable Energies

Climate change and the finite nature of fossil fuels have brought about the need for alternative energy solutions. In 2015, 196 sovereign nations pledged to join efforts in holding the global average temperature below 2 °above pre-industrial levels, in the Paris Agreement, a legally binding treaty signed at the UN Climate Change Conference (COP21) [7]. This was to be achieved by reducing greenhouse gas emissions. In [8], the authors propose a new limit of 1.5 °, stating that further temperature increments might be cause for public concern.

To fight climate change and its prejudicial effects, the European Union launched several goals in 2018. One of these goals was for each member state to achieve 32 % of its total energy consumption from renewable sources by 2030. Each member state was also to submit a National Plan for Energy and Climate. In 2020, Portugal presented the Plano Nacional Energia e Clima (PNEC 2030), with a 47 % gross final renewable energy consumption goal [9]. The trajectory proposed in the PNEC 2030 for renewable energy consumption can be seen in table 2.1, adapted from [9].

Table 2.1: Trajectory and Portugal’s contribution to the European Union’s 2030 binding goal.

	2020	2022	2025	2027	2030
Renewable energy in gross final energy consumption	31%	34%	38%	41%	47%

In the same document, table 2.2, adapted from [9], shows it is expected that 9 GW in photovoltaic energy capacity will be installed by the year 2030. Of the three energy sources with the highest desirable capacity, solar, wind, and hydroelectric, solar has the largest margin for growth, 7 GW, when compared to wind and hydroelectric’s 3.6 and 1.2 GW margins, respectively.

By the end of 2022, REN’s yearly technical data report showed that 1.89 GW of photovoltaic solar energy capacity had been installed [10].

Table 2.2: Perspective for the evolution of the installed capacity for electricity per source in Portugal until 2030.

(GW)	2020	2025	2030
Hydroelectric	7	8.2	8.2
from pumping	2.7	3.6	3.6
Wind Power	5.4	6.8	9.3
Wind Power onshore	5.4	6.7	9.0
Wind Power offshore	0.03	0.1	0.3
Solar (Photovoltaic) - total	2	6.6	9.0
Solar (Photovoltaic) - centralised	1.5	5.8	7.0
Solar (Photovoltaic) - not centralised	0.5	0.8	2.0
Concentrated Solar Thermal Energy	0	0.1	0.3
Biomass	0.4	0.4	0.5
Other renewable energy sources	0.03	0.06	0.1
Geothermal	0.03	0.03	0.06
Waves	0.001	0.03	0.07
Coal	1.8	0	0
Natural Gas	3.8	3.8	2.8-3.8
Fuel/Diesel	0.4	0.3	0.3
Total	20.8	26.3	30.5-31.5

2.2 Photovoltaic Power Plant Overview

A utility-scale photovoltaic solar system connected to the electric grid comprises solar panels, inverters, a power conditioning unit and equipment for grid connection [11].

In this section, solar panels and inverters will be looked into in more detail.

2.2.1 Photovoltaic Modules

One of the major challenges associated with photovoltaic technologies is the non-linear nature of its current-voltage (I-V) characteristics [12].

According to [13], in semiconductors, there are bonded electrons (valence electrons) and free electrons (conduction electrons). Photovoltaic cells, the most basic element in PV modules, work under the photoelectric principle. This principle states that under specific conditions, an electron can absorb the energy from a photon. This absorbed energy can force electrons to migrate from the valence to the conduction band.

Silicon (Si) is one of the most common materials used for semiconductors, with 4 valence electrons, and it can be doped with small amounts of Boron (B) and Phosphorus (P), with 3 and 5 valence electrons, respectively. In Boron-doped Silicon, there will be one leftover Si electron that cannot bond, resulting in a positive (p) charge. Positive charges are commonly named holes, seen as they are the absence of an electron. In Phosphorus-doped Silicon, one free Phosphorus electron

will join the conduction band, a higher energy level where electrons are free to move, resulting in a negative (n) charge.

A diode is made with p-type and n-type doped materials in a p-n junction or depletion zone. This junction is naturally polarised, meaning free electrons in the n-type layer do not cross into the p-type layer. When a diode is connected to a source with the p-zone connected to the higher potential end and the n-zone to the lower potential end, the diode is considered to be in direct polarisation and will conduct. If the diode is connected to the source with the p and n zones switched, it is called reverse polarisation and will not conduct.

A solar cell can be likened to a diode. The absorbed photons will originate electron-hole pairs in the depletion zone. When repeated many times, a disequilibrium will occur in the diode, with more net accumulated positive charges in the p-zone and more negative charges in the n-zone. This imbalance leads to a difference of potential.

Several cells are used to make up a single PV module. However, the module can be represented in the same way as a cell, which means, with a single diode equivalent circuit, as seen in Fig. 2.1, adapted from [13].

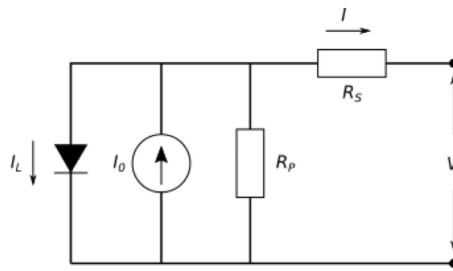


Figure 2.1: PV module equivalent circuit.

From the equivalent circuit, a panel can then be mathematically modelled with equation 2.1.

$$I = I_L - I_0(T) \left(e^{\frac{eV + R_s I}{m k T}} - 1 \right) - \frac{V + R_s I}{R_p} \quad (2.1)$$

I and V stand respectively for the produced current and voltage by the cells/module, I_L is the theoretically produced current from light absorption, $I_0(T)$ is the temperature-dependent reverse bias saturation current, e is the electron charge, R_s is the series resistance of the cell, R_p is the parallel resistance of the cell, m is the diode ideality factor and k is the Boltzmann constant.

This model also informs the I-V characteristic curve of a cell, as seen in Fig. 2.2, adapted from [13]. The figure shows important parameters to characterise a cell or module. These are short-circuit current (I_{sc}), open circuit voltage (V_{OC}), the maximum power point current (I_{mpp}) and the maximum voltage power point (V_{mpp}).

2.2.2 Inverters

The inverter plays a crucial role in grid-connected solar systems, as stated in [14]. When designing a photovoltaic solar power plant, the inverter selection determines the system's DC voltage. After

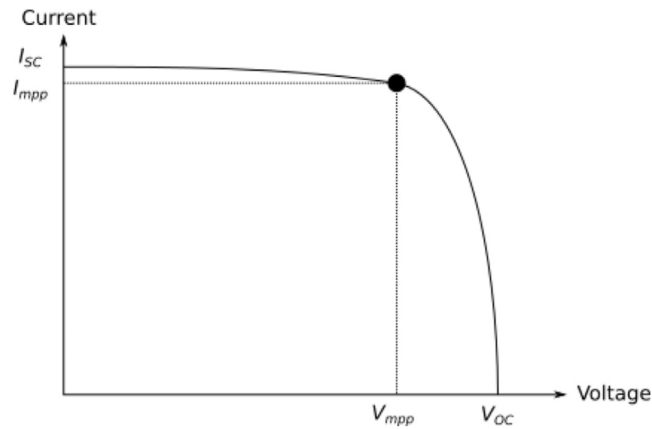


Figure 2.2: Characteristic I-V curve of a cell.

that, the selection of a solar module model follows. The inverter is tasked with converting the direct current produced by the PV modules into 50 Hz alternating current, which is the grid's frequency.

Inverters ensure the plant begins operation at the right time of day, as soon as the cells produce enough power. The ideal operating point of a solar power system is influenced by the level of solar radiation and module temperature, which both fluctuate during the day. To ensure maximum efficiency, inverters are equipped with Maximum Power Point Trackers (MPPTs) that allow them to adjust their operation to the optimal working point. Moreover, inverters have a built-in protection mechanism that stops their operation in case of system faults, be it on the grid or solar generator side.

Inverters have data loggers and measurement computers that constantly capture production data such as power, voltage, current, and other parameters. This information is transmitted and stored regularly through a serial interface and can be accessed later on a computer for analysis and further use.

2.3 Partial Shading Related Failures and Losses

2.3.1 Hot Spot Formation

Shading or other types of failure in a cell or cells might lead to lower short-circuit current values when compared to other modules in the same string [14]. When a cell is fed a current higher than the one it is currently generating, it will become reverse-biased. In this case, it will dissipate power instead of generating it, behaving as a resistance. This phenomenon, see Fig. 2.3, adapted from [13], also called hot spot, can lead to the burning of the shaded cell.

Bypass diodes are a common yet imperfect solution used to solve the hot spot problem, for they require the cells to achieve the diode's operating point, otherwise, it will not divert the current [15].

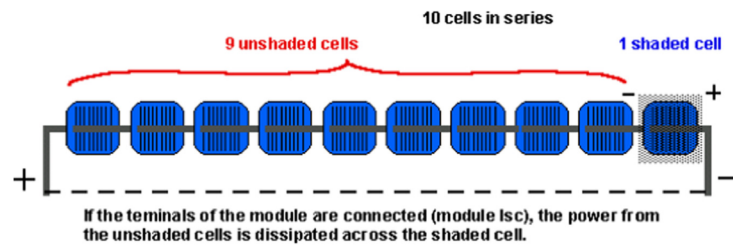


Figure 2.3: Hotspot creation

2.3.2 Maximum Power Point Tracking Difficulties

Tracking the maximum power point becomes challenging when a photovoltaic (PV) array experiences partial shading or operates in rapidly changing insolation conditions. This difficulty arises due to the emergence of two or more local maximum power points. Non-uniform insolation conditions or partial shading of PV modules can lead to multiple local maximum operating points on the PV power–voltage (P–V) characteristic. Consequently, the operating point of the PV system may tend to converge towards a local maximum that does not correspond to the actual maximum point on the P–V curve [16].

2.4 Maintenance

Modern technologies relating to the Internet of Things, sensing technology, and artificial intelligence have evolved maintenance strategies. This evolution has seen a shift from Reactive Maintenance (RM) to Preventive Maintenance (PM) to Predictive Maintenance (PdM) [17].

RM is a method of maintenance that is carried out only after equipment failure has occurred. This approach often results in a significant lag in restoring equipment to its operational state and incurs high reactive repair costs.

PM, on the other hand, involves carrying out maintenance according to a pre-planned schedule based on time or process iterations to prevent equipment breakdowns. However, this method may result in unnecessary maintenance and high prevention costs.

To achieve the best balance between these two approaches, PdM is performed based on an online estimate of the "health" of the equipment. This method allows for timely pre-failure interventions and enables the maintenance frequency to be as low as possible to prevent unplanned RM without incurring costs associated with doing too much PM.

As an example to illustrate PV system maintenance actions, the maintenance planning standard for power plants mentioned in [18], the following tasks are included: inspecting system and wiring components, structural attachments, and sealing against adverse weather conditions; cleaning and removing moisture and debris around arrays; grass-cutting; conducting electrical tests; and replacing or repairing damaged or failed system components. It becomes clear that the majority of these measures are preventative.

There are various strategies for cleaning modules, which may involve different cleaning methods, such as mechanical or static, wet or dry, automatic or manual. Additionally, different brushes or cloths and chemical cleaners may be used. However, frequent cleaning can lead to high maintenance costs, particularly in areas where water is scarce and labour is expensive [19].

Predictive Maintenance in photovoltaic systems is an interesting application for Machine Learning [17], this topic will be further developed in Chapter 3.

Chapter 3

Related Work

In this chapter, some proposed approaches for assessing shading losses or overall detecting anomalies in PV systems were looked into. Topics such as machine learning and neural networks were covered.

3.1 Knowledge Based Approaches

Traditional methods for Predictive Maintenance (PdM) are based on expert knowledge and deductive reasoning. These methods include the use of expert systems and model-based reasoning for fault diagnosis and prognosis [17].

There are different types of reasoning methods employed in knowledge-based PdM, including Ontology, Rule, and Model-based approaches. Ontology provides a way to integrate, share, and reuse context knowledge of a system. However, other reasoning methods should be integrated with ontology to achieve PdM.

Rule-based approaches are useful when there is a lot of experience but not enough details to develop accurate quantitative models. For example, large industrial plants that are difficult to model, where linear approximation models may result in large errors. However, a rule-based system may have difficulties in dealing with new faults and requires complete knowledge to build a reliable rule-based system.

Model-based approaches apply when accurate mathematical models can be constructed from physical systems. However, explicit mathematical models may not be feasible for many complex systems. Additionally, changes in structural dynamics and operating conditions can affect the mathematical model, making it impossible to model all real-life conditions.

A methodology was proposed in [2] to assess annual shading losses due to partial shading. This approach estimates the theoretical production without shading, disregarding string data from strings that have a higher than 5 % discrepancy from a chosen reference unshaded string.

To detect five specific PV system anomalies, daytime zero-production, low maximum production, daytime shading, sunrise/sunset shading, and sub-optimal orientation, 5 different algorithms were tailor-made [20]. A scale for anomaly severity for each anomaly was also proposed. Despite

being successful in detecting anomalies, these algorithms relied on several heuristic parameters and could not be employed in higher latitudes because of day-time offset constraints, where the day becomes too short for the algorithm to do anomaly detection.

3.2 Machine Learning Based Approaches

In our daily activities, Machine Learning has become an integral part of our experiences, seamlessly enhancing efficiency, convenience, and security across various domains [21]. Whether engaging with banks, shopping online, or navigating social media, ML algorithms continually adapt to our preferences and lifestyles, making our interactions more streamlined and personalised. The potential for Machine Learning applications in advancing renewable energy development is virtually boundless. Numerous stakeholders in this sector are exploring inventive solutions to enhance the efficiency of their systems.

According to [21], Common issues in photovoltaic (PV) systems can be automatically identified through machine learning (ML) approaches, primarily employing three distinct methodologies. Firstly, there is the analysis of string/panel current or voltage, including measurements at the inverter, utilising external factors such as environmental variables. This is often achieved through established ML methods like Artificial Neural Networks (ANN), Fuzzy Logic (FL), Decision Tree (DT), and Random Forest (RF).

The second approach involves image analysis, predominantly utilising infrared (IFR) images captured by Unmanned Aerial Vehicles (UAVs). Deep Learning (DL), particularly with various types of Convolutional Neural Networks (CNNs), has proven effective in this context.

The third methodology revolves around clustering-based techniques that detect anomalies using unlabelled data. Key methods encompass k-nearest Neighbour (kNN), one-class Support Vector Machine (1-SVM), as well as more recent algorithms such as Isolation Forest (IS) or Local Outlier Factor (LOF).

Within the literature, the predominant focus revolves around four categories of faults in photovoltaic systems: short circuit (SC), open circuit (OC), partial shading (PS), and abnormal ageing. The prevalent approach for addressing these faults involves employing a Multi-Layer Perceptron Artificial Neural Network (MLP ANN). This network typically inputs the current or voltage parameters associated with the string/array/panel. The most commonly considered variables include voltage at Maximum Power Point (VMPP), current at Maximum Power Point (IMPP), OC voltage (VOC), and SC current (ISC). These models are frequently augmented by environmental variables like ambient and module temperature and solar irradiance at the panel level.

Crucially, these models necessitate a labelled dataset and primarily rely on discerning differences between predicted and measured system performance.

The following sections will provide a more detailed explanation of various ML models.

3.2.1 Artificial Neural Networks (ANN)

Artificial Neural Network (ANN) is a computing paradigm that aims to imitate the human brain's performance, such as learning from experiences, generalising from similar situations, and making judgements based on states [17].

Dr Jürgen Schmidhuber defines Artificial Neural Networks in his work [22] as complex grids with fundamental units known as neurons. These neurons will take input data from sensors or previous neuron layers and pass it on to the next layer through a web of weighted connections. The weights of the connections are real-valued and can be modified, determining the behaviour of the network.

A Feedforward Neural Network (FNN) is a basic and commonly used type of artificial neural network. It consists of several layers of artificial neurons arranged in a sequence. In a feedforward neural network with L hidden layers, the network is described by $L + 1$ weight matrices ($\mathbf{W}_0, \mathbf{W}_1, \dots, \mathbf{W}_L$) and $L + 1$ bias vectors ($\mathbf{b}_1, \mathbf{b}_2, \dots, \mathbf{b}_{L+1}$). Together, these weight matrices and bias vectors form the parameter vector θ , which completely defines the function that the network computes. When given an input $\mathbf{x}$, the feedforward neural network processes it to produce an output $\mathbf{z}$ [23]:

Algorithm 1 Feedforward Neural Network Computation

```

1: Set  $z_0 \leftarrow \mathbf{x}$ 
2: for  $i \leftarrow 1$  to  $L + 1$  do
3:    $\mathbf{x}_i \leftarrow \mathbf{W}_{i-1}\mathbf{z}_{i-1} + \mathbf{b}_i$ 
4:    $z_i \leftarrow e(\mathbf{x}_i)$ 
5: end for
6: Output  $\mathbf{z} \leftarrow z_L$ 

```

In this algorithm, $e(\cdot)$ stands for a non-linear activation function, such as element-wise sigmoid, arctangent, and Rectified Linear Unit (ReLU) functions [24].

To train a feedforward neural network, we must minimise an objective function E (a loss function or cost function), which measures the difference between the network's predictions and target values. Examples of objective functions include Mean Squared Error (MSE) for regression tasks and Cross-Entropy Loss for classification tasks [25].

The process of minimising the objective function and finding the optimal parameters of the network is done using optimisation algorithms such as Stochastic Gradient Descent (SGD) [23].

A Recurrent Neural Network (RNN) is a neural network designed to process sequential data by maintaining a hidden state that captures information about the previous inputs. This allows RNNs to exhibit temporal dynamic behaviour, making them suitable for tasks involving sequences, such as time series analysis, natural language processing, and speech recognition [26].

RNNs are parameterised with three weight matrices and three bias vectors $[\mathbf{W}_{hv}, \mathbf{W}_{hh}, \mathbf{W}_{oh}, \mathbf{b}_h, \mathbf{b}_o, \mathbf{h}_0]$ whose concatenation θ completely describes the RNN (see Fig. 2.3). Given an input sequence $(\mathbf{v}_1, \dots, \mathbf{v}_T)$ (which we denote by $\mathbf{v}_1^T$), the RNN computes a sequence of hidden states $\mathbf{h}_1^T$ and a sequence of outputs $\mathbf{z}_1^T$ using the following algorithm:

Algorithm 2 Recurrent Neural Network Computation

```

1: for  $t$  from 1 to  $T$  do
2:    $\mathbf{net}_t \leftarrow \mathbf{W}_{hv}\mathbf{v}_t + \mathbf{W}_{hh}\mathbf{h}_{t-1} + \mathbf{b}_h$ 
3:    $\mathbf{h}_t \leftarrow e(\mathbf{net}_t)$ 
4:    $\mathbf{o}_t \leftarrow \mathbf{W}_{oh}\mathbf{h}_t + \mathbf{b}_o$ 
5:    $\mathbf{z}_t \leftarrow g(\mathbf{o}_t)$ 
6: end for

```

Here, $e(\cdot)$ and $g(\cdot)$ are the non-linear activation functions for the hidden and output layers, respectively, and $\mathbf{h}_0$ is the initial hidden state.

The loss function of the RNN is typically the sum of losses at each time step:

$$L(\mathbf{z}, \mathbf{y}) = \sum_{t=1}^T L(\mathbf{z}_t; \mathbf{y}_t)$$

The gradients of the RNN are efficiently computed using the Backpropagation Through Time (BPTT) algorithm. A single iteration of gradient descent using backpropagation calculates changes to all weights w_i in proportion to $\frac{\partial E}{\partial w_i} = \sum_t \frac{\partial E}{\partial \mathbf{net}_t} \frac{\partial \mathbf{net}_t}{\partial w_i}$, as described in Algorithm 3. Each weight w_i is associated with a real-valued variable Δ_i , which is initially set to 0. [22]:

Algorithm 3 Backpropagation Error Signal Computation

```

1: for  $t = T, \dots, 1$  do
2:   to compute  $\frac{\partial E}{\partial u_t}$ , initialize real-valued error signal variable  $\delta_t$  by 0;
3:   if  $x_t$  is an input event then
4:     continue with next iteration
5:   end if
6:   if there is an error  $e_t$  then
7:      $\delta_t := x_t - d_t$ 
8:   end if
9:   Add to  $\delta_t$  the value  $\sum_{k \in \text{out}_t} w_{v(t,k)} \delta_k$ ; (this is the elegant and efficient recursive chain rule application collecting impacts of  $\mathbf{net}_t$  on future events)
10:  Multiply  $\delta_t$  by  $f'_t(\mathbf{net}_t)$ 
11:  for all  $k \in \text{int}$  do
12:    Add to  $\Delta w_{v(k,t)}$  the value  $x_k \delta_t$ 
13:  end for
14:  Change each  $w_i$  in proportion to  $\Delta_i$  and a small real-valued learning rate
15: end for

```

RNNs, however, have a vanishing gradient problem, making training them on long data sequences challenging. First identified in [27], gradient vanishing, also known as the *long time lag problem*, shaped DL research in the 1990s and 2000s, according to [22]. The vanishing gradient problem occurs when the gradients of the loss function concerning the weights in a neural network become very small during backpropagation. This makes it difficult for the network to learn and update the weights, especially for long-range dependencies in sequential data. As a result, the

network struggles to capture and utilise information from earlier time steps, hindering its ability to learn effectively.

This limitation led to the development of the Long Short-Term Memory (LSTM) network, which addresses the shortcomings of traditional RNNs by introducing a memory cell and gating mechanisms to better capture long-range dependencies in sequential data [26]. LSTMs will be further explored in Chapter 4.

3.2.2 Support Vector Machine (SVM)

A Support Vector Machine (SVM) is a supervised machine learning technique that is useful when the mathematical relationship between the input and output variables is complex or unknown. It is also effective when the number of interdependent factors is high, making it difficult to obtain a precise mathematical model [28].

In a classification task, samples are usually assumed to have positive and negative classes. An SVM training algorithm constructs a model that assigns new examples to one of these two categories. The model is a non-probabilistic binary linear classifier [17].

Suppose there is a set of training data $(x_1, y_1), \dots, (x_l, y_l) \subset X \times R$, where X represents the space of input patterns (for example, $X = R_d$). These patterns could be exchange rates of a currency measured on different days, along with their corresponding econometric indicators. In ϵ -SV regression [28], the goal is to find a function $f(x)$ that deviates by no more than ϵ from the actual targets for all the training data and, at the same time, is as flat as possible. We only focus on errors less than ϵ and do not accept any deviation larger than this. This approach may be helpful to ensure that we don't lose more than ϵ money while dealing with exchange rates [29].

3.2.3 Decision Trees (DT)

Decision Tree (DT) is a supervised method used for classification and regression. It predicts a target variable's value (i.e., label or class) by learning simple decision rules from the data features. A DT comprises a root node, branches, internal nodes and leaves. A path from the root node through the internal nodes to a leaf node represents a classification with different conditions of the systems or components. Each leaf node represents a class label for classification or a response for regression.

To build a DT model, one should identify the most important input variables/features and then split instances at the root node and subsequent internal nodes into two or more categories based on the status of such variables [17].

3.2.4 Auto-Encoder (AE)

Autoencoders (AEs) are a deep learning model. AEs are used to learn generative models of data. They take input data and convert it into an abstract representation, which is then converted back into the original format using the encoder function. The AE is specifically trained to encode the input into a representation that can be used to reconstruct the input. Essentially, the AE approximates

the identity function during this process. One key advantage of the AE is that it can continuously extract useful features during propagation and filter out any useless information. Additionally, since the input vector is transformed into a lower-dimensional representation during the coding process, the learning process becomes more efficient [30].

3.2.5 Convolutional Neural Networks (CNN)

CNNs (Convolutional Neural Networks) are a type of deep learning neural network that can extract local features from input data and combine them layer by layer to generate high-level features [17]. A typical CNN structure consists of an input layer, a convolution layer, a pooling layer, and a fully connected layer.

The input layer can be presented in a two-dimensional manner, such as a time-frequency spectrum, or in a one-dimensional manner, such as time series data. The input data can be represented as $X \in R^{A \times B}$, where A and B are the dimensions of the input data.

In the convolution layer, the convolution kernel (filter) convolutes the input data from the previous layer through a set of weights and composes a feature output, generally called a feature map. The output of the convolutional layer can be calculated as:

$$Y_{cn} = f(X * W_{cn} + b_{cn}) \quad (3.1)$$

Here, " * " represents a convolution operator, cn denotes the number of convolution filters, W_{cn} is the weight matrix of the cn-th filter kernel, b_{cn} is the kernel bias, and f is an activation function such as rectified linear units (ReLU).

The pooling layer performs a sampling operation to reduce model parameters and retain effective information. This helps to avoid overfitting to some extent and improves training speed. The most commonly used pooling layer is the max-pooling layer, which extracts the maximum value from Y_{cn} as follows:

$$P_{cn} = \max_{SM \times N}(Y_{cn}) \quad (3.2)$$

Here, $SM \times N$ is a scaling matrix of pooling, and M and N are the dimensions of S.

After several combinations of convolution and pooling layers, multiple fully connected layers follow. These layers convert the matrix in the filter to a column or a row. Finally, a classification or regression layer can be added to achieve specific aims.

3.2.6 Research Findings in Fault Detection and Model Comparison

In [3], the authors propose an algorithm that can predict equipment faults several days or weeks before they occur. To achieve this, an ANN estimates the power output by simulating the behaviour of the panels plus inverters group for the current conditions. The inverters will communicate their output (voltage and current) in real-time, which is then compared to the simulation values. If the

growing gap between production and simulation is a trend, adequate maintenance procedures are issued.

An approach for unsupervised fault detection has been proposed [31]. This approach combines the benefits of model-based techniques with the classification capacity of clustering algorithms, such as one-class SVM. The approach uses the one-diode-based simulation model to describe the nominal behaviour of a PV array by using real environmental measurements as inputs. It then applies one-class SVM to the generated residuals from the simulation model to detect faults.

An approach was presented in [4] in which three neural network architectures, 1D Convolutional Neural Networks, Single Headed Long Short Term Memory (LSTM) and multi-headed LSTM have their performance in anomaly detection tested. These algorithms must be able to detect anomalies and classify IV curves from collected data into baseline/normal, partially soiled and cracked. The authors found that multi-headed LSTM and 1D CNN architectures produced accuracies greater than 99% on average, the best of the three. While 1D CNN was marginally more accurate, multi-headed LSTM took half the time to train.

Three machine learning schemes were also tested in [5]. AutoEncoder LSTM (AE-LSTM), Facebook-Prophet and Isolation Forrest were used to evaluate and distinguish healthy from abnormal PV system behaviour. It was observed that only AE-LSTM correctly identified all of the anomalies and the healthy signal with the other two models identifying the anomalies but labelling positive points as anomalies.

Chapter 4

Methodologies and Implementations

The previously mentioned case study is a state-of-the-art photovoltaic power plant in southern Portugal equipped with DC/AC inverters capable of recording historical data, which is then used for monitoring purposes. These measurements are taken every ten minutes, with each inverter registering current and voltage data. At the plant's weather station, equipment registers the local irradiance and the temperature inside and outside solar cells.

When developing the models, the first data analysis phase consisted of understanding and processing said data, identifying patterns and translating it into a format that could later be used in ML applications. During this phase, it was necessary to model the system's expected power to calculate the difference between expected and measured power. This difference would later be used as input for the models. This analysis would shape how the models acquire the data, which will be reported in this section. The expected power was calculated based on solar irradiance measurements taken at the power plant weather station and solar panel characteristics such as peak power and temperature coefficients, which are present in the panel datasheet [32].

The second phase was the conception of the algorithms themselves and the usage of the data processed in phase one. The K-Means-based approach was developed with MatLab. The LSTM-based approach was developed with Python in VS Code and visualised with MatLab. The algorithms and the challenges met during their development will be discussed in subsections 4.1 and 4.2.

The real datasets from the case-study plant were used in the data analysis phase. The available data was recorded by two inverters from areas with different slopes. The data was recorded during July of 2023 and January of 2024.

When analysing and pre-processing the datasets, it became apparent that there were instances when the measured irradiance would be zero while the inverters still registered power production. Likewise, zero output in the inverters with irradiance being registered also happened. The distance between the inverters and the weather station means clouds could shade them at different intervals. Equipment faults could also cause this phenomenon.

To model the expected power production without partial shading, the following equation, based on an equation found in [33] was utilised:

$$P_{exp} = n \cdot \frac{P_{max} \cdot I}{I_{STC}} \cdot \frac{100 + Coef_{voc} \cdot (T - T_{STC})}{100} \quad (4.1)$$

This equation calculates the expected power output (P_{exp}) of a PV system by considering the number of panels (n), the maximum power output of each panel under standard conditions (P_{max}), the actual irradiance (I), temperature conditions (T), and the temperature coefficient of the open circuit voltage ($Coef_{voc}$). The adjustments for irradiance and temperature ensure that the expected power output reflects the real operating conditions rather than the Standard Test Conditions (STC). STC typically means an irradiance (I_{STC}) of 1000 W/m² and a cell temperature (T_{STC}) of 25°C. The P_{max} , I_{STC} , T_{STC} and $Coef_{voc}$ values were taken from the solar panel datasheet [32] and can be seen in Table :

After calculating the expected power, it was possible to compare the results with the measured power.

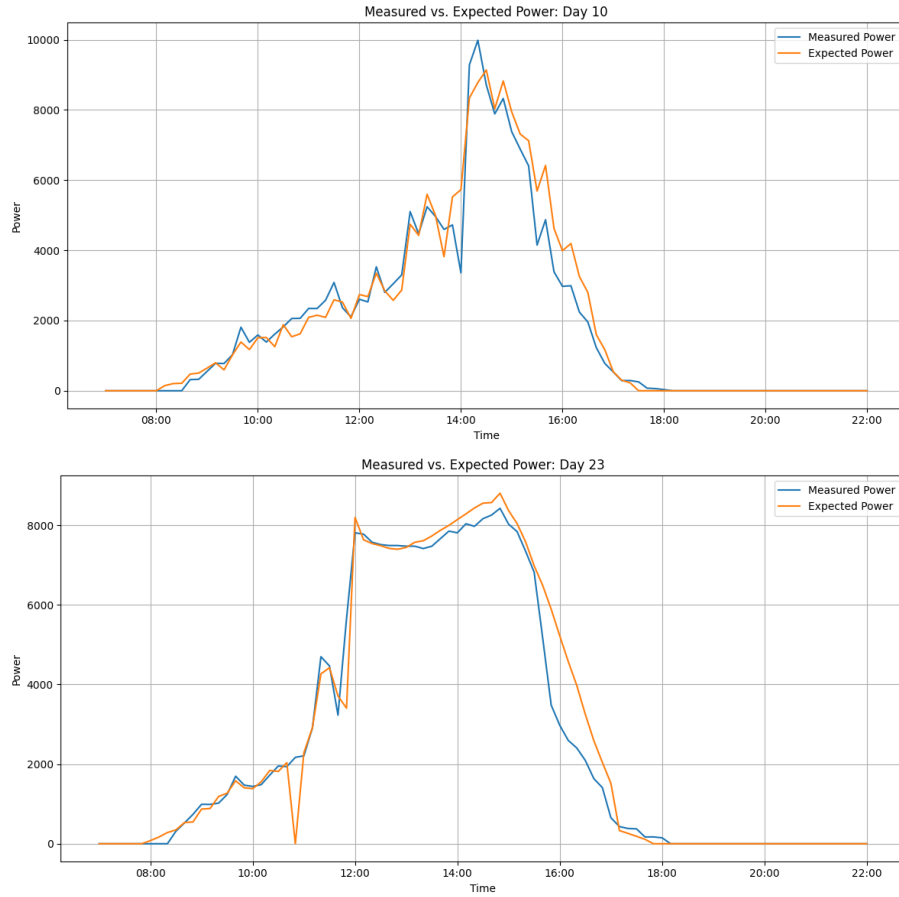


Figure 4.1: Measured vs Expected Power for string 2 on days 10 (TOP) and 23 (BOTTOM) in January

Figure 4.1 shows a noticeable gap between the expected and measured power from 3 to 5 pm over the two days. This phenomenon was recognised for this string in 15 of the 31 days in the month. The other strings exhibited similar shading profiles.

We calculated and plotted the difference between the measured and expected power values to convert this data into a convenient format for the models.

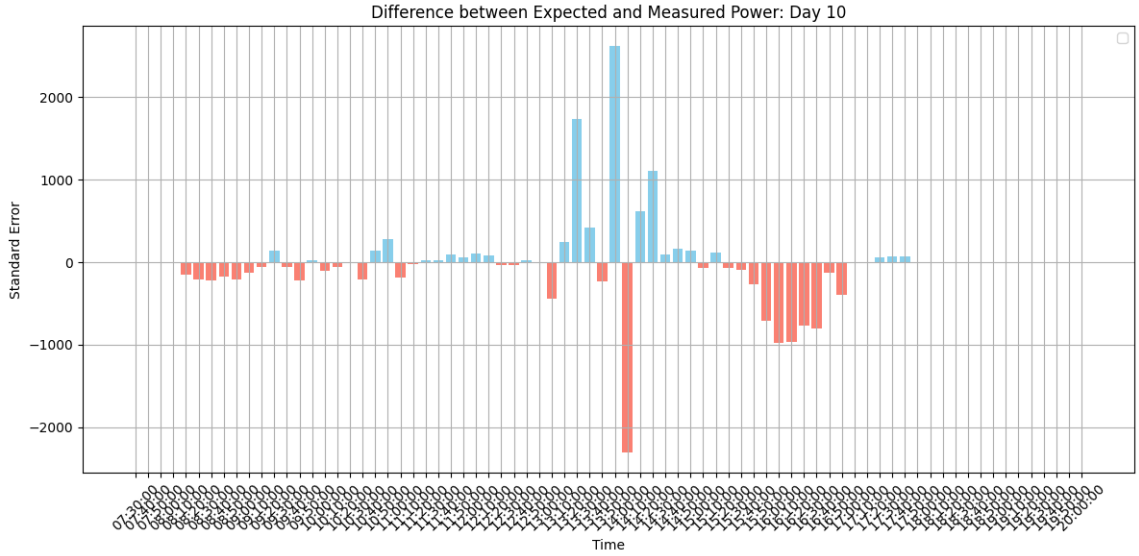


Figure 4.2: Difference between Expected and Measured Power: Day 10, string 2

The previously mentioned gap between the expected and measured power on day 10 becomes evident in graph 4.2.

4.1 Methodologies - K-Means

Clustering algorithms organise data into groups based on its perceived characteristics. The key difference between data clustering (unsupervised learning) and classification or discriminant analysis (supervised learning) lies in the absence of category information. Clustering seeks to uncover structure within the data, making it an exploratory process [34].

Let $X = \{x_i \mid i = 1, \dots, n\}$ be a set of n points in d -dimensional space that are to be clustered into K clusters, $C = \{c_k \mid k = 1, \dots, K\}$. The K-means algorithm seeks to partition the points such that the squared error between each point and the mean of its assigned cluster is minimised. Let ℓ_k be the mean of cluster c_k [35]. The squared error between ℓ_k and the points in cluster c_k is defined as:

$$J(c_k) = \sum_{x_i \in c_k} \|x_i - \ell_k\|^2$$

The objective of K-means is to minimise the total squared error across all K clusters:

$$J(C) = \sum_{k=1}^K \sum_{x_i \in c_k} \|x_i - \ell_k\|^2$$

According to [36], the K-means algorithm can be described as follows:

Algorithm 4 Iterative Partitional Clustering

- 1: **Step 1:** Select an initial partition with K clusters.
- 2: Repeat steps 2 through 5 until the cluster membership stabilises.
- 3: **Step 2:** Generate a new partition by assigning each pattern to its closest cluster center.
- 4: **Step 3:** Compute new cluster centers as the centroids of the clusters.
- 5: **Step 4:** Repeat steps 2 and 3 until an optimum value of the criterion function is found.
- 6: **Step 5:** Adjust the number of clusters by merging and splitting existing clusters or by removing small or outlier clusters. =0

A first tentative version of the K-Means approach was made with Python in VS-Code. However, the time-series nature of the data proved to be challenging. The clustering algorithm performed clustering based on the proximity of the timestamps as a sequence, prioritising the standard error value and the error's periodicity to a lesser extent.

To solve this problem, the data was converted into a matrix and plotted in a 3-D mesh with MatLab, as seen in the following example for string 15:

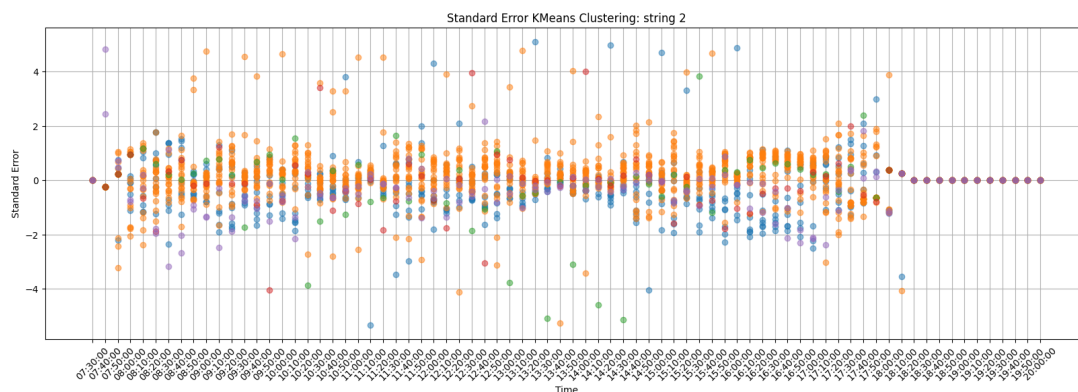


Figure 4.3: Standard Error KMeans Clustering: string 7

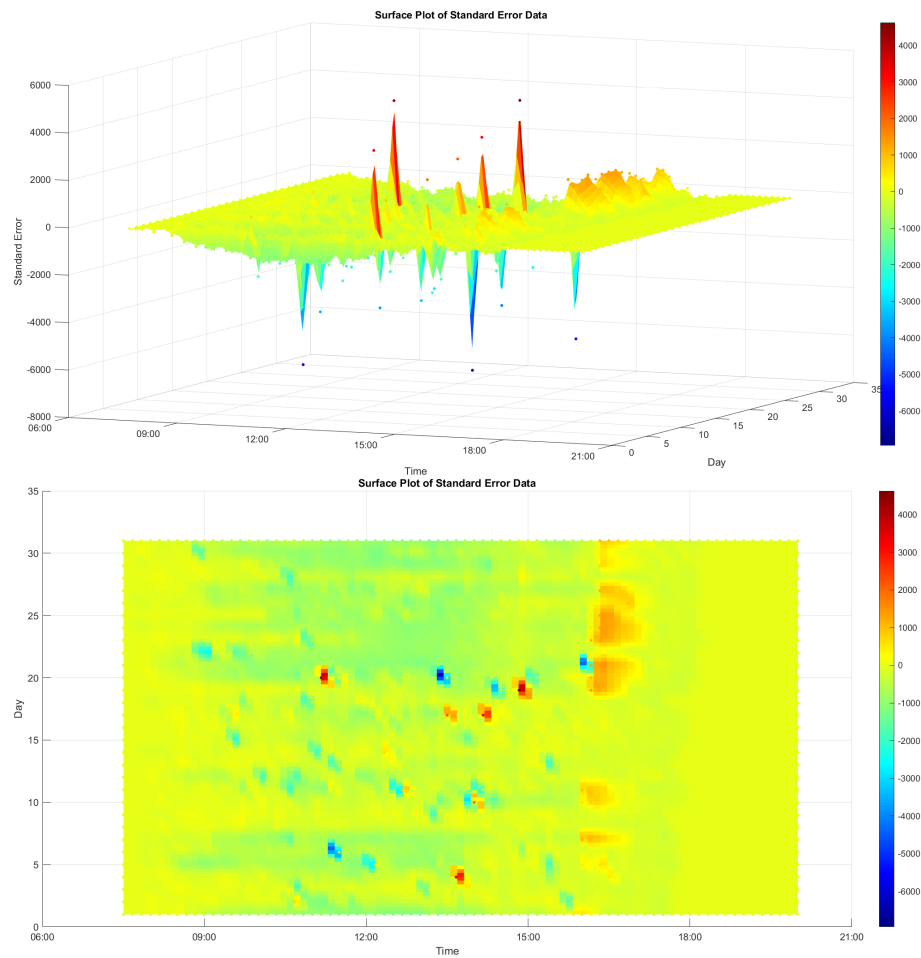


Figure 4.4: Difference between expected and measured power in January for string 15 (TOP: 3D Perspective View of Surface Plot, BOTTOM: Top-down View of 3D Surface Plot)

Within the depicted diagram 4.4, it is observable that the losses occurring between 15:00 and 17:00 are consistent, suggesting that these shading losses can be attributed to the design of the plant, as anticipated. The values along the Z-axis have been inverted for improved clarity and interpretation of the outcomes. The K-Means algorithm was applied to the string. The results can be seen in Figure 4.5.

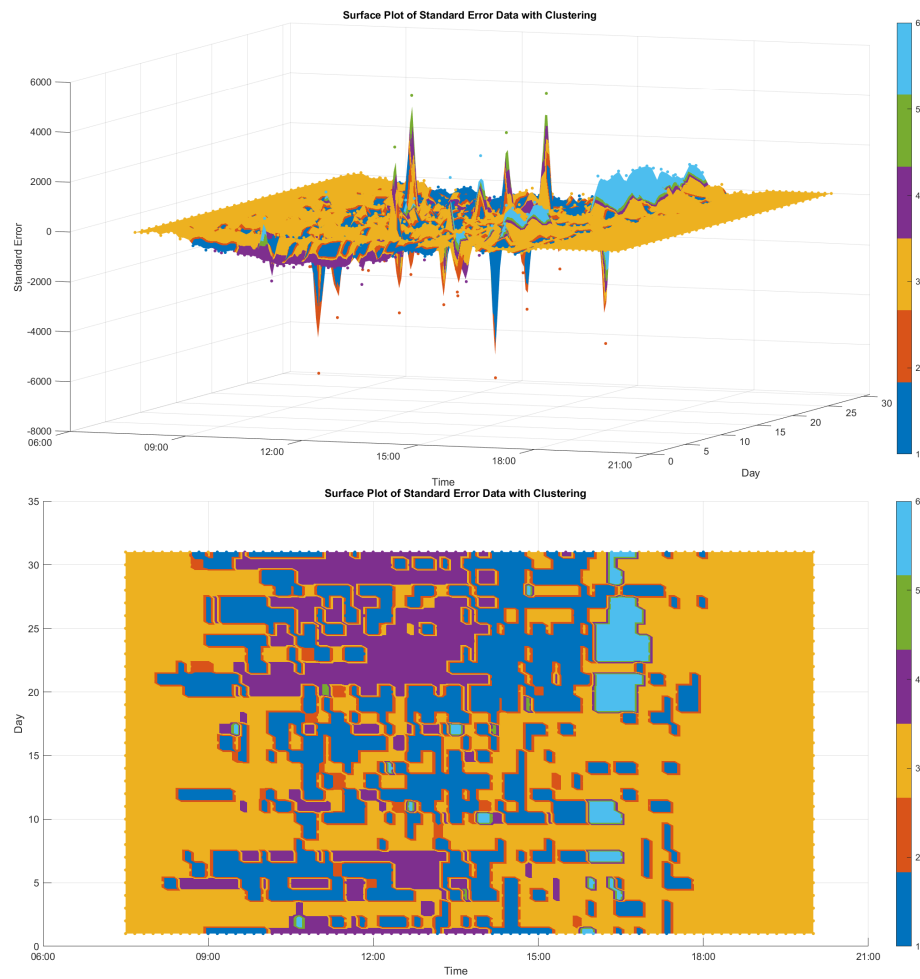


Figure 4.5: K-means clustering applied to the loss profile of string 15 in January (TOP: 3D Perspective View of Surface Plot, BOTTOM: Top-down View of 3D Surface Plot)

Each colour in the graph indicates a different cluster, with light blue, cluster 6, belonging to partially shaded timestamps. Table 4.1 shows each cluster's total power in W in the column Total Power, the number of timestamps belonging to each cluster in the column Count. The asterisk (*) indicates that the values are contained to the time interval between 15:00 and 18:00. According to this model, this string lost 46.6 kW in January.

Table 4.1: Total power and timestamp count per cluster for string 15 in January.

Cluster	Total Power in kW	Timestamp Count	Total Power in kW*	Timestamp Count *
1	-234.2	560	-49.9	135
2	-71.2	24	-12.2	4
3	-68.0	1390	-33.3	387
4	-286.8	313	-49.7	5
5	15.6	4	0.0	0
6	53.8	65	46.6	58

This model could group the timestamps according to their similarity by their power difference value, time of day and day of the month. Despite the model correctly identifying partially shaded periods, there were also false positives. These can be easily found in Figure ?? as there are light blue clusters before 15:00. Since no labelled data is available for this case, verifying false positives after 15:00 is impossible.

4.2 Methodologies - LSTM

From the previously conducted literature research, LSTM neural networks consistently had the best results out of several machine learning models. As mentioned in Section 3.2.1, LSTM NNs are a kind of RNN. RNNs are particularly suited for applications where the output depends on prior inputs, such as time series analysis. The LSTM architecture was one of the solutions devised to solve the vanishing gradient problem.

According to [37], an LSTM memory cell, denoted by c_j , is built around a central linear unit with a fixed self-connection known as the Constant Error Carousel (CEC). This self-connection helps maintain constant error flow and mitigates the vanishing gradient problem often encountered in RNNs.

An LSTM memory cell, denoted by c_j , is built around a central linear unit with a fixed self-connection known as the Constant Error Carousel (CEC). This self-connection helps maintain constant error flow and mitigates the vanishing gradient problem often encountered in RNNs.

To control the flow of information and avoid conflicts, LSTMs use three gate units:

- **Input Gate** (i_j): Controls the addition of new information to the memory cell.
- **Forget Gate** (f_j): Determines how much of the current memory cell value should be forgotten.
- **Output Gate** (o_j): Regulates the output of the memory cell's value.

The input gate i_j manages the flow of information to the memory cell c_j by controlling the input connections w_{cji} , thus preventing input weight conflicts. Similarly, the output gate o_j manages the flow of information from the memory cell by controlling the output connections, preventing conflicts in the output weights. Essentially, the input gate decides when to keep or overwrite the information in the memory cell, while the output gate determines when to access the memory cell's information and when to protect other units from being affected by it.

Given the context of the equations provided:

$$y^{out_j}(t) = f_{out_j}(net_{out_j}(t))$$

$$y^{in_j}(t) = f_{in_j}(net_{in_j}(t))$$

where:

$$net_{out_j}(t) = \sum_u w_{out_ju} y^u(t-1)$$

$$net_{in_j}(t) = \sum_u w_{in_j,u} y^u(t-1)$$

Algorithm 5 LSTM Cell Operation

- 1: **Input:** Previous cell state c_{t-1} , input x_t
 - 2: **Output:** Cell state c_t , output h_t
 - 3: **Initialize:** Forget gate f_t , Input gate i_t , Output gate o_t , Cell candidate $\tilde{c}_t$
 - 4: COMPUTE $f_t \leftarrow \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$ {Forget gate activation}
 - 5: COMPUTE $i_t \leftarrow \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$ {Input gate activation}
 - 6: COMPUTE $\tilde{c}_t \leftarrow \tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$ {Cell candidate}
 - 7: UPDATE CELL STATE $c_t \leftarrow f_t \cdot c_{t-1} + i_t \cdot \tilde{c}_t$ {New cell state}
 - 8: COMPUTE $o_t \leftarrow \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$ {Output gate activation}
 - 9: COMPUTE $h_t \leftarrow o_t \cdot \tanh(c_t)$ {New hidden state}
 - 10: **Return:** $c_t, h_t = 0$
-

Using the PyTorch framework, an LSTM NN model was developed and tested in Python for the second part of our methodology.

4.2.1 Data Preprocessing

Unlike the previously implemented unsupervised model, the biggest challenge faced when developing this supervised machine learning algorithm was the need for labelled data.

To solve this problem, the first model was used to generate labels. In the results relating to string 7, the labels for all timestamps were converted to 0, except for the ones between 15:00 and 18:00. All labels different from 6 were also converted to 0 to have only two classes and those with values equal to 6 were changed to 1. Class 0 meant the timestamp was not part of a partially shaded period, and class 1 meant the time stamp was included in a shaded period.

We found it necessary to standardise the features. The StandardScaler function from the Sickit-Learn library was used to standardise the data. This function standardises each feature by normalising the samples within said features. In this case, the features would be each day of the month, and each sample would be the measurements taken between 7:30 and 20:00, resulting in 31 samples with 76 features each. The function calculates each sample's mean (μ) and standard deviation (σ) and then standardises the features according to the following expression:

$$X_{std} = \frac{x - \mu}{\sigma} \quad (4.2)$$

Standardising the data will prevent features from having variances magnitudes higher than others, causing them to become dominant in the objective function and prevent the estimator from learning from other features correctly [38].

Before applying the mentioned function, this classifier often produced false positives, as seen in Figure 4.6.

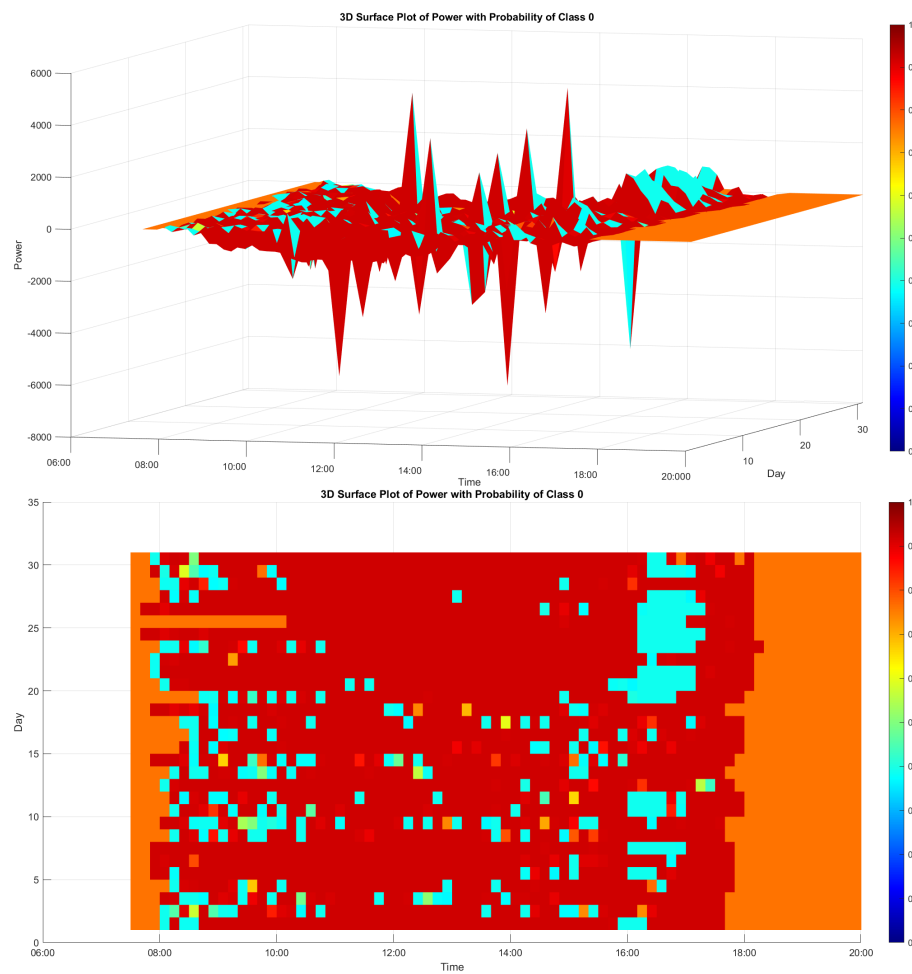


Figure 4.6: LSTM classifier applied to the loss profile of string 15 in January, before data standardisation, showcasing false positives

(TOP: 3D Perspective View of Surface Plot, BOTTOM: Top-down View of 3D Surface Plot)

In this example, there were a total of 143 timestamps that were falsely classified as partially shaded before 15:00. For comparison, after normalisation, the best model, see Figure 4.12, misclassified 9 timestamps. Another common problem before standardisation was that the model would struggle with proper probability attribution, often labelling shaded periods as having a high probability of being unshaded and unshaded periods as having a high probability of being shaded, as seen in Figure 4.7.

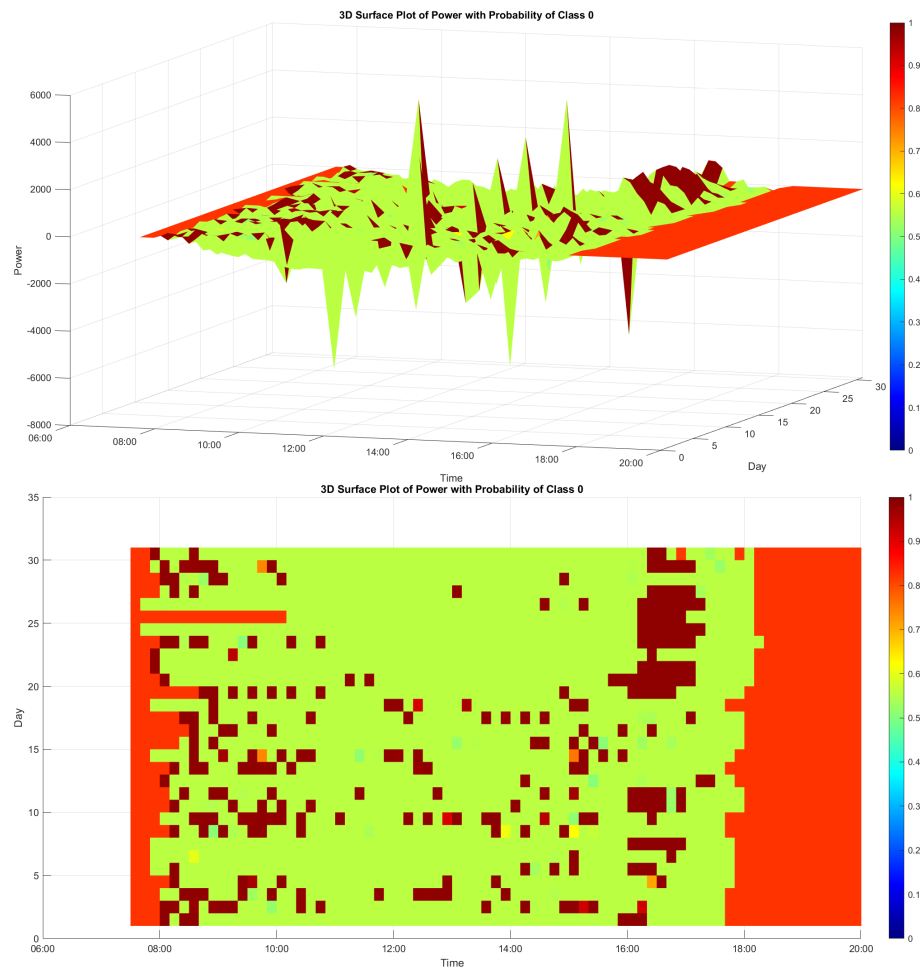


Figure 4.7: LSTM classifier applied to the loss profile of string 15 in January, before data standardisation, showcasing poor probability attribution.

(TOP: 3D Perspective View of Surface Plot, BOTTOM: Top-down View of 3D Surface Plot)

Neural Networks do not favour unbalanced data sets when training. Unbalanced data sets are those where some classes have disproportionally higher expression than the rest.

Before balancing the training data sets data, the resulting models would consistently fail to attribute the correct probability intervals to the shaded periods when applied to real string data.

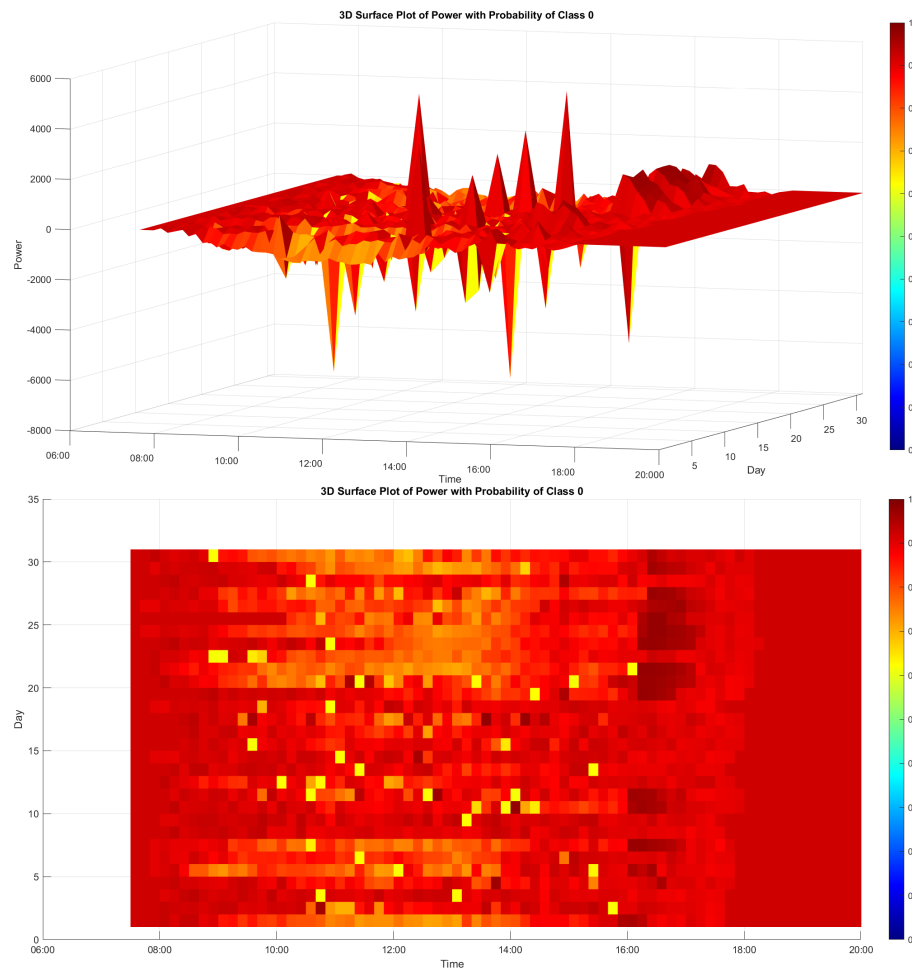


Figure 4.8: LSTM classifier applied to the loss profile of string 15 in January before balancing the training data set.

(TOP: 3D Perspective View of Surface Plot, BOTTOM: Top-down View of 3D Surface Plot)

A visual analysis of Figure 4.8 shows that the model could identify shaded periods but would attribute these timestamps a high probability of being shaded, above 0.8. This can be confirmed in Table 4.2, where the positive sum of power losses is concentrated in the high probability of class 0 levels.

Table 4.2: Total power per probability of Class 0 interval for string 15 in January

Probability Interval	Sum of Standard Error in kW	Timestamp Count	Sum of Standard Error in kW *	Timestamp Count *
0.0-0.1	0.0	0.0	0.0	0.0
0.1-0.2	0.0	0.0	0.0	0.0
0.2-0.3	0.0	0.0	0.0	0.0
0.3-0.4	49.3	33.0	28.7	26.0
0.4-0.5	13024.165	20.0	11698.161	18.0
0.5-0.6	9200.789	22.0	7620.049	18.0
0.6-0.7	12609.707	104.0	4472.295	31.0
0.7-0.8	-116819.090	1411.0	-50523.240	374.0
0.8-0.9	-507137.705	746.0	-46776.879	88.0
0.9-1.0	-50989.600	20.0	-6681.918	3.0

Since the partial shading affected periods are contained between 15:00 and 18:00, the data from strings 7, 2 and 8 were filtered accordingly. The matrix relative to a string's monthly production had dimensions 31 by 76. To achieve this shape, the 25 by 31 matrices that resulted from filtering were concatenated and padded. Figure 4.9, at the top, shows the mapped labels used to train the models before balancing the training dataset. After balancing, the label mapping became as seen in Figure 4.9, at the bottom. Label 0 timestamps are coloured in black, and label 1 in white.

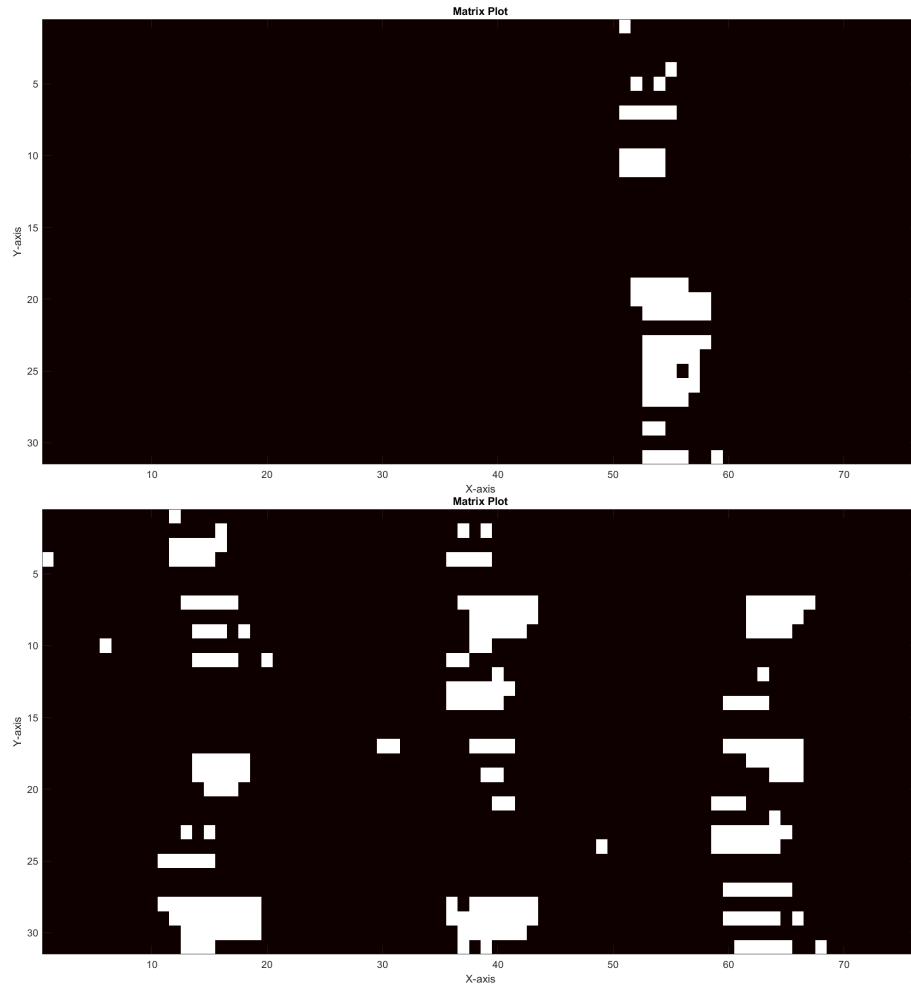


Figure 4.9: TOP: Mapped labels before training data set balancing. BOTTOM: Mapped labels after training data set balancing.

Balancing the training datasets increased labels with value 1 from 2.8% to 9.25%. This 6.45% increase in positive labels allowed our models to do proper probability attributions, as seen in Figure 4.12.

4.2.2 Neural Network Architecture

It was then necessary to define the LSTM model architecture. According to [3], it is general knowledge that using a wider hidden layer instead of several hidden layers diminishes the complexity of

the model and processing times.

To determine the batch size, number of neurons and learning rate, examples found in literature [3] were used as a basis. These values were then tuned through a trial-and-error process, validated by comparisons with the K-means model, and visual analysis of the resulting graphs. In graph analysis, the following features were favourably weighted towards the architecture: clearly defined shaded periods and correct attribution of class probability to the timestamps to shaded periods, with a higher probability of class 0 for non-shaded periods and a lower probability of class 0 for shaded periods. Class 0 was chosen for data visualisation out of convenience.

The loss curve is calculated by recording the loss value at the end of each training epoch during the model's training loop. In each epoch, the model processes batches of training data, computing the predicted outputs and calculating the loss using the cross-entropy loss function. This loss measures the difference between the predicted outputs and the actual labels. After each batch, the optimiser updates the model parameters to minimise this loss. The loss value for the epoch is then appended to a list, which is subsequently used to plot the loss curve, visually representing the model's training progress over time and showing how well it is learning.

The loss curve is expected to show a downward trend over time, indicating that the model's performance improves as the training progresses. A decreasing loss curve signifies that the model is learning and the difference between its predictions and the actual labels is getting smaller. Ideally, the curve should steadily decline and eventually plateau, showing that the model has converged and further training yields diminishing returns in reducing the loss. Anomalies, such as a consistently high or increasing loss, might indicate issues like insufficient learning, overfitting, or problems with the data or model architecture. This curve was utilised as an indicator during the NN architecture engineering phase. The loss curve for the final version of the architecture presented in Figure 4.10 meets the previous descriptions of the expected curve.

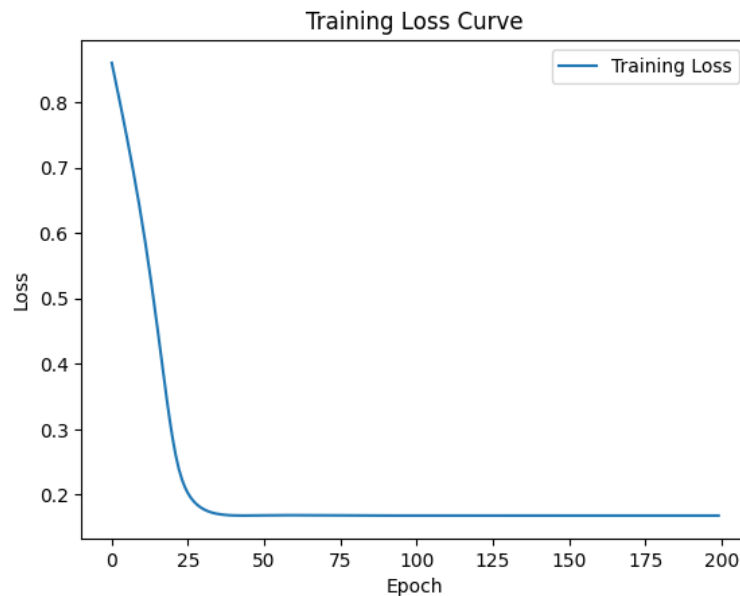


Figure 4.10: Final Training Loss Curve

When defining the batch size, there was a trade-off between variety in results and consistency, with smaller batch sizes favouring the first and higher sizes favouring the latter. Iterations were made with the batch size ranging from 1 to 64. The size that better aligned with the required features and offered the best results was 32. The model with a smaller batch size, 16 samples per batch, would not converge in a stable plateau, as seen in Figure 4.11. Larger batch sizes lead to the results converging to undesirable values, where the shaded periods were never assigned probabilities of being shaded higher than 50%.

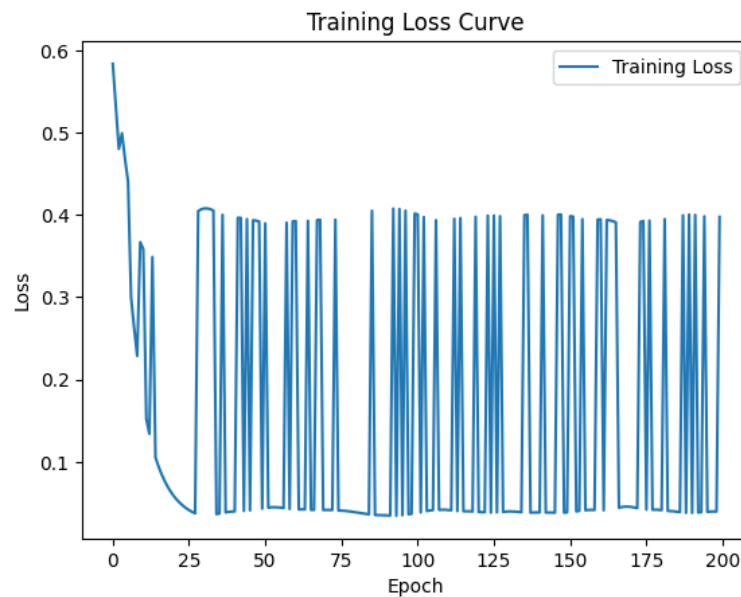


Figure 4.11: Training Loss Curve with Batch Size 16

The exact process was followed to determine the number of neurons and learning rate. The neurons were incremented in steps of 1 from 1 to 20. The optimal solution that did not lead to under or overfitting while clearly defining shaded periods was 4 neurons. The learning rate was incremented from 0.001 to 0.01 in steps of 0.001, with the final value for the model being 0.008.

Given the random initialisation of the weights and the stochastic nature of training in neural networks, this architecture produced different models for each training iteration for string 15's data set. Figure 4.12 and Table 4.3 show the resulting graphs and recorded values for the model that better satisfied the previously established guidelines. Once again, the asterisk (*) indicates that the values are contained to the time interval between 15:00 and 18:00

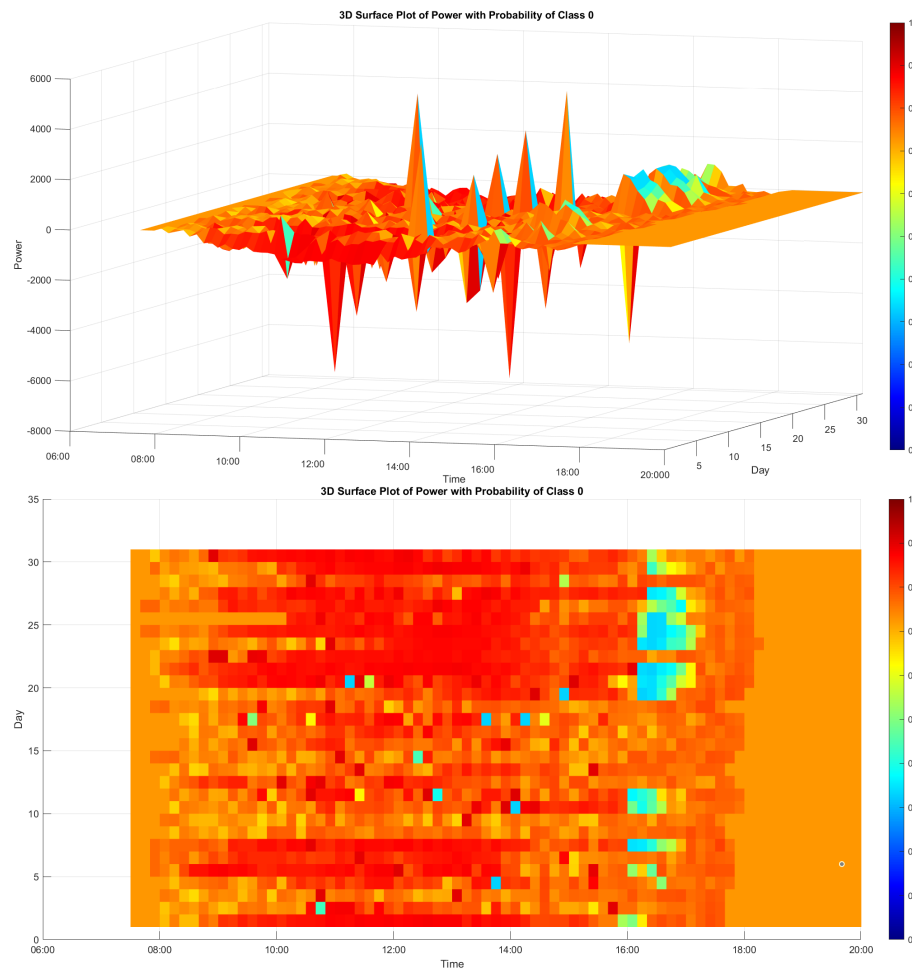


Figure 4.12: LSTM classifier applied to the loss profile of string 15 in January.
(TOP: 3D Perspective View of Surface Plot, BOTTOM: Top-down View of 3D Surface Plot)

Table 4.3: Total power per probability of Class 0 interval for string 15 in January

Probability Interval	Sum of Standard Error in kW	Timestamp Count	Sum of Standard Error in kW *	Timestamp Count *
0.0-0.1	0.0	0.0	0.0	0.0
0.1-0.2	0.0	0.0	0.0	0.0
0.2-0.3	0.0	0.0	0.0	0.0
0.3-0.4	49.3	33.0	28.7	26.0
0.4-0.5	13.0	20.0	11.7	18.0
0.5-0.6	9.2	22.0	7.6	18.0
0.6-0.7	12.6	104.0	4.5	31.0
0.7-0.8	-11.7	1411.0	-50.5	374.0
0.8-0.9	-50.7	746.0	-46.8	88.0
0.9-1.0	-51.0	20.0	-6.7	3.0

As with the previous models, all the timestamps classified as partially shaded, in this case, with a probability of class 0 below 0.5, before 15:00, are guaranteed to be false positives. According to this model, the assessed partial shading losses, between 15:00 and 18:00 and with a probability of class 0 of 0.5 or below, were 40.4 kW.

Chapter 5

Results and Conclusions

5.1 Validation and Results

5.1.1 K-Means-based Models Results

Despite the K-means algorithm not being stochastic, initialising the K cluster centres is random, introducing stochasticity into the process. As such, the results of 20 runs of this program were recorded and studied to understand its behaviour.

This program randomly attributes a number to the clusters, which means, for example, that what is shown as cluster 6 in Figure 4.5 could have any number between 1 and 6 in a different run. To align the cluster numbers between the 20 runs, it was necessary to apply the Kuhn–Munkres algorithm, also known as the Hungarian method [39]. The mentioned method is a combinatorial optimisation algorithm that solves assignment problems, such that the total cost is minimised or the total profit is maximised. It is useful in cases where it is necessary to match two sets of items optimally. In this case, cluster labels assigned by the algorithm can change across different runs even if the clusters themselves are stable. This variability makes it difficult to compare clusters directly. The Hungarian method optimises the assignment problem by minimising the total cost and maximising the overlap between clusters across different runs. It provides a way to consistently label clusters across multiple runs of the K-means algorithm, enabling meaningful comparisons and evaluations of clustering stability and performance.

After applying the mentioned method, it was possible to compare the results from the recorded runs visually. Figure 5.1 shows two examples where cluster 1 pertains to the shaded periods. The graphs for each of the 20 runs can be consulted in the annex.

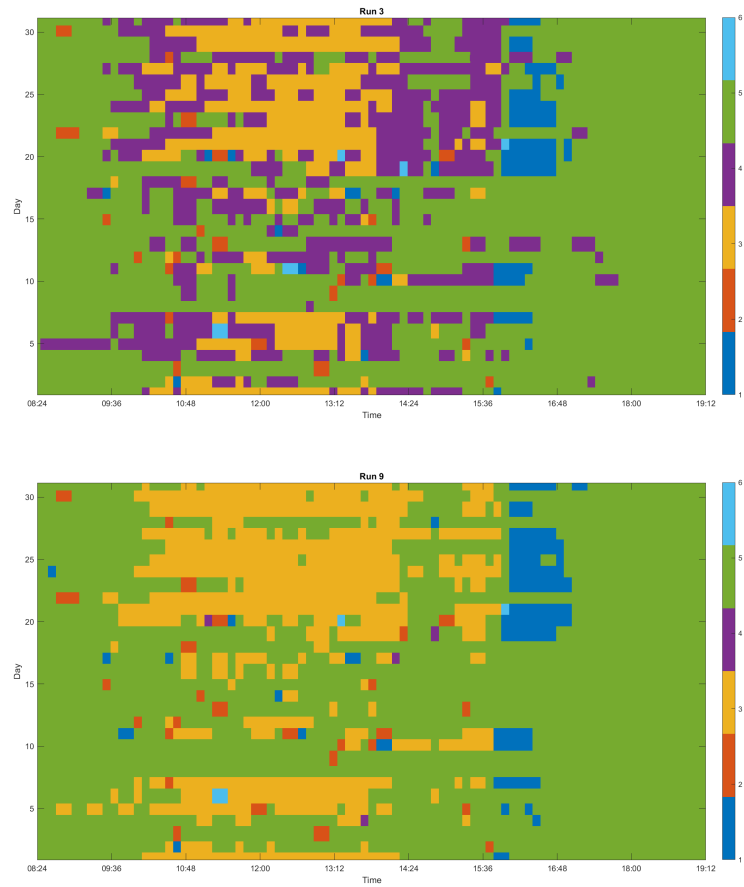


Figure 5.1: Runs 3 (TOP) and 9 (BOTTOM) of K-means clustering and Hungarian algorithm applied to the loss profile of string 15 in January.

The distribution over time of timestamps with label 1 can be seen in Figure 5.2:

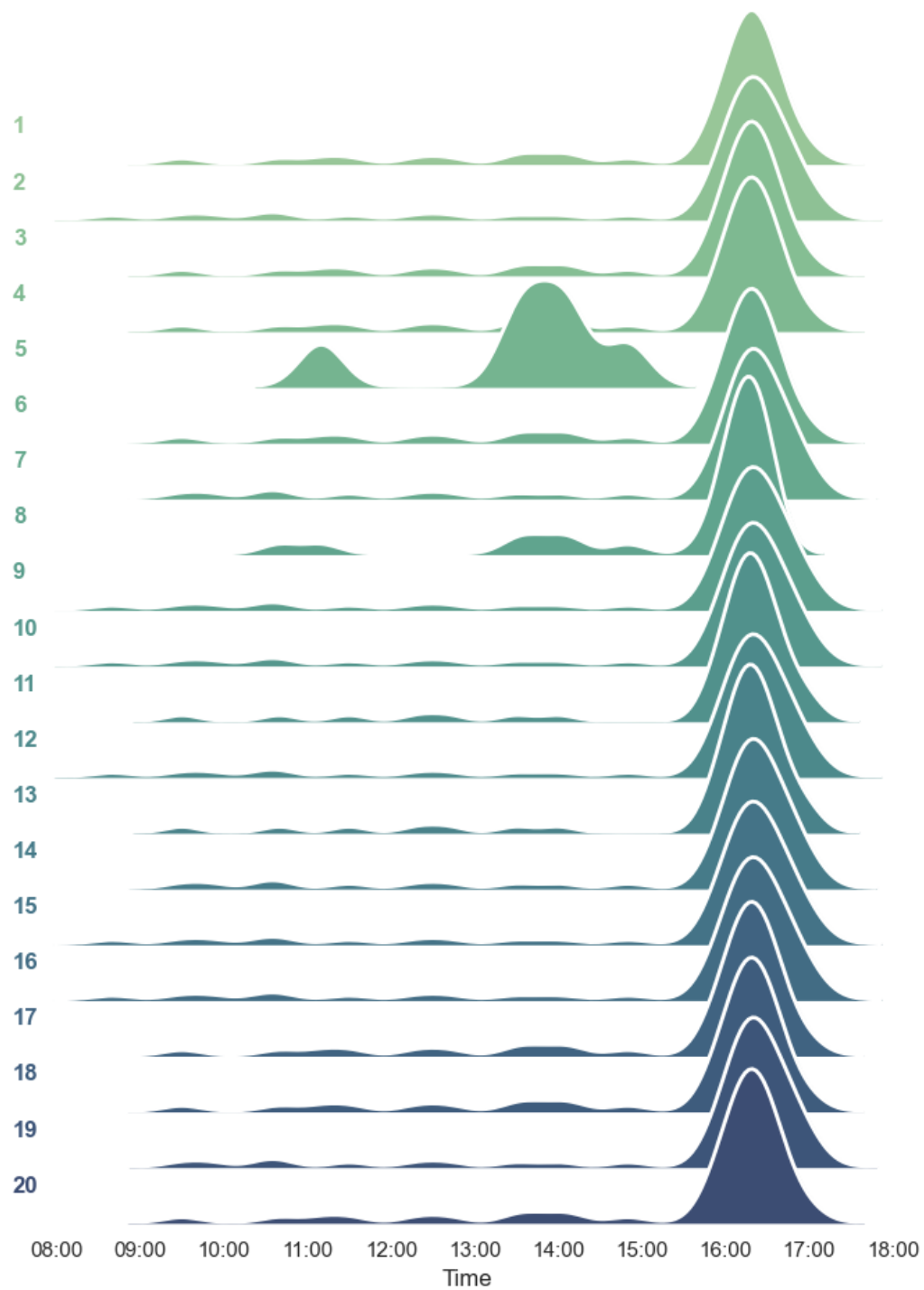


Figure 5.2: Cluster 1 distribution over time, by model

Analysing figure 5.2 allowed us to easily identify that run 5 had 100% of the label 1 timestamp before 15:00. This model could classify timestamps as partially shaded in the correct period in 19 out of 20 runs. An average of 19.706% of the classified timestamps were false positives.

Developing this model allowed us to confirm one of the most well-known disadvantages of K-means clustering. It is necessary to input the total number of clusters before the clustering occurs, demanding trial and error determination of this number and previous knowledge of the data set.

5.1.2 LSTM-based Models Results

Previously, it was mentioned that the training process is stochastic. This architecture was initialised and trained 20 times with the combined dataset from strings 7, 2 and 8, and each model was then used to process the data from string 15 in January. Figure 5.3 illustrates the distribution of 558 timestamps between 15:00 and 18:00, with each model's probability values represented by density plots. The density plots provide a representation of the distribution of probability values. From this figure, it is possible to infer that three models out of 20 could classify timestamps as partially shaded. Model 1, seen in Figure 4.12, classified 44 timestamps (40.4 W) as partially shaded, whereas Model 2 classified 11 timestamps (14.1 W) as partially shaded and Model 9 timestamps as partially shaded (10.6 W).

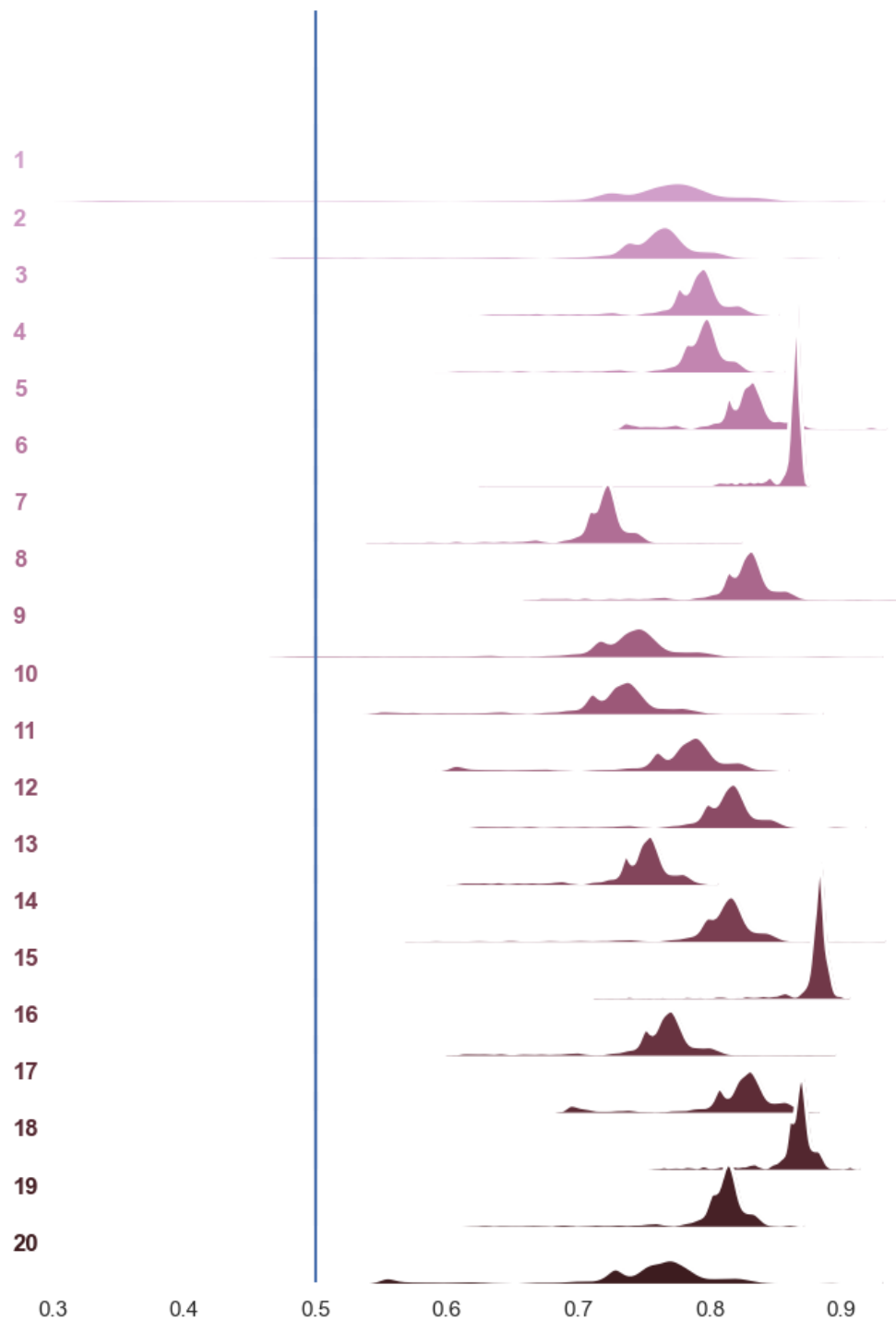


Figure 5.3: Probability distribution, by model

5.1.3 Model Results Comparison

On January 15, 2024, CJR Renewables conducted an experiment between 14:00 and 16:00 to assess the impact of shadow growth on panel productivity. The increase in shadow presence over

the panels during this time frame was noted to have ranged from 1 cm to 10 cm on average. Furthermore, the study revealed a 20% decrease in expected production during this period.

With this reference and having calculated the difference between measured and expected power for that period, the results for the models were expressed in Table 5.1

Table 5.1: Results for the studied models. False positives are only counted before 15:00

	Total Expected Power in kW	Calculated Losses in kW	Losses as percentage	False positives
Reference	N/A	N/A	20.0%	N/A
Calculated	301.6	53.1	17.6%	N/A
K-Means	301.6	46.6	15.4%	7
LSTM Model 1	301.6	40.4	13.4%	9
LSTM Model 2	301.6	14.1	4.7%	6
LSTM Model 9	301.6	10.6	3.5%	9

The K-Means-based model was the closest to the reference values from these results while having fewer false positives than the best LSTM Model. It must be noted that when weighting these models, the K-Means-based approach holds all the disadvantages mentioned in Section 4.1. The LSTM models, on the other hand, do not require previous knowledge of the dataset and do not demand choosing a cluster number.

5.2 Conclusions and Future Work

The main objective of this work was to develop and study machine learning models capable of assessing losses due to partial shading. Two distinct approaches were shown: an unsupervised machine learning approach based on K-Means clustering and a supervised machine learning approach aided by unsupervised ML and based on LSTM Neural Networks. Assessing and attributing these losses to this fault can be key to future penalty negotiations in O&M. The models described in this paper could do so by drawing similarities and finding patterns in the differences between the expected and measured power. This diagnostic tool might also be valuable for informing all parts involved in photovoltaic power plant conception and engineering phases. This project allowed us to study and gain valuable insight into machine learning, respective applications and good practices, and important concepts related to photovoltaic energy.

The models were shown to have validation errors. The biggest challenge for the LSTM models was the lack of labelled data for this case. With more data for training purposes, we can achieve better future results and improve the means of validation.

Future work might include finer tuning of hyperparameters through the development of parameter optimisation models to improve the current capabilities of this system.

We also aim to experiment with new and different neural network architectures, for example, using an AutoEncoder in conjunction with the existing LSTM.

Additional features, such as the estimation of how the registered partial shading periods affect the ageing of the solar cells and calculating the estimated losses in revenue and equipment substitution, would be interesting from a commercial application standpoint.

References

- [1] E. Ortega, G. Aranguren, M. Saenz, R. Gutierrez, and J. Jimeno, “Study of Photovoltaic Systems Monitoring Methods,” in *2017 IEEE 44th Photovoltaic Specialist Conference (PVSC)*. Washington, DC: IEEE, Jun. 2017, pp. 643–647. [Online]. Available: <https://ieeexplore.ieee.org/document/8366523/>
- [2] H. Zsiborács, L. Zentkó, G. Pintér, A. Vincze, and N. H. Baranyai, “Assessing shading losses of photovoltaic power plants based on string data,” *Energy Reports*, vol. 7, pp. 3400–3409, Nov. 2021. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S2352484721003176>
- [3] M. De Benedetti, F. Leonardi, F. Messina, C. Santoro, and A. Vasilakos, “Anomaly detection and predictive maintenance for photovoltaic systems,” *Neurocomputing*, vol. 310, pp. 59–68, Oct. 2018. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S0925231218305484>
- [4] M. W. Hopwood, T. Gunda, H. Seigneur, and J. Walters, “Neural Network-Based Classification of String-Level IV Curves From Physically-Induced Failures of Photovoltaic Modules,” *IEEE Access*, vol. 8, pp. 161 480–161 487, 2020. [Online]. Available: <https://ieeexplore.ieee.org/document/9186596/>
- [5] M. Ibrahim, A. Alsheikh, F. Awaysheh, and M. Alshehri, “Machine Learning Schemes for Anomaly Detection in Solar Power Plants,” *Energies*, vol. 15, no. 3, p. 1082, Feb. 2022. [Online]. Available: <https://www.mdpi.com/1996-1073/15/3/1082>
- [6] C. Deline, B. Sekulic, J. Stein, S. Barkaszi, J. Yang, and S. Kahn, “Evaluation of Maxim module-Integrated electronics at the DOE Regional Test Centers,” in *2014 IEEE 40th Photovoltaic Specialist Conference (PVSC)*. Denver, CO, USA: IEEE, Jun. 2014, pp. 0986–0991. [Online]. Available: <http://ieeexplore.ieee.org/document/6925080/>
- [7] “The Paris Agreement | UNFCCC.” [Online]. Available: <https://unfccc.int/process-and-meetings/the-paris-agreement>
- [8] D. I. Armstrong McKay, A. Staal, J. F. Abrams, R. Winkelmann, B. Sakschewski, S. Loriani, I. Fetzer, S. E. Cornell, J. Rockström, and T. M. Lenton, “Exceeding 1.5°C global warming could trigger multiple climate tipping points,” *Science*, vol. 377, no. 6611, p. eabn7950, Sep. 2022. [Online]. Available: <https://www.science.org/doi/10.1126/science.abn7950>
- [9] “Resolução do conselho de ministros n.º 53/2020, de 10 de julho,” pp. pp 2 – 158, Sep. 2020. [Online]. Available: <https://diariodarepublica.pt/dr/detalhe/resolucao-conselho-ministros/53-2020-137618093>

- [10] REN, “Technical data 2022.” [Online]. Available: https://www.ren.pt/media/noho4j45/dados_te-cnicos_ren_22.pdf
- [11] B. Shiva Kumar and K. Sudhakar, “Performance evaluation of 10 MW grid connected solar photovoltaic power plant in India,” *Energy Reports*, vol. 1, pp. 184–192, Nov. 2015. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S2352484715000311>
- [12] H. Patel and V. Agarwal, “Maximum Power Point Tracking Scheme for PV Systems Operating Under Partially Shaded Conditions,” *IEEE Transactions on Industrial Electronics*, vol. 55, no. 4, pp. 1689–1698, Apr. 2008. [Online]. Available: <http://ieeexplore.ieee.org/document/4439206/>
- [13] L. Hernández-Callejo, S. Gallardo-Saavedra, and V. Alonso-Gómez, “A review of photovoltaic systems: Design, operation and maintenance,” *Solar Energy*, vol. 188, pp. 426–440, Aug. 2019. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S0038092X19305912>
- [14] A. Luque and S. Hegedus, Eds., *Handbook of photovoltaic science and engineering*. Hoboken, NJ: Wiley, 2003.
- [15] C. Olalla, M. Hasan, C. Deline, and D. Maksimović, “Mitigation of Hot-Spots in Photovoltaic Systems Using Distributed Power Electronics,” *Energies*, vol. 11, no. 4, p. 726, Mar. 2018. [Online]. Available: <http://www.mdpi.com/1996-1073/11/4/726>
- [16] N. A. Ahmed and M. Miyatake, “A novel maximum power point tracking for photovoltaic applications under partially shaded insolation conditions,” *Electric Power Systems Research*, vol. 78, no. 5, pp. 777–784, May 2008. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S0378779607001368>
- [17] Y. Ran, X. Zhou, P. Lin, Y. Wen, and R. Deng, “A Survey of Predictive Maintenance: Systems, Purposes and Approaches,” Dec. 2019, arXiv:1912.07383 [cs, eess]. [Online]. Available: <http://arxiv.org/abs/1912.07383>
- [18] G. D. Lorenzo, R. Araneo, M. Mitolo, A. Niccolai, and F. Grimaccia, “Review of O&M Practices in PV Plants: Failures, Solutions, Remote Control, and Monitoring Tools,” *IEEE Journal of Photovoltaics*, vol. 10, no. 4, pp. 914–926, Jul. 2020. [Online]. Available: <https://ieeexplore.ieee.org/document/9103493/>
- [19] J. Chen, G. Pan, J. Ouyang, J. Ma, L. Fu, and L. Zhang, “Study on impacts of dust accumulation and rainfall on PV power reduction in East China,” *Energy*, vol. 194, p. 116915, Mar. 2020. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S0360544220300220>
- [20] P. Branco, F. Gonçalves, and A. C. Costa, “Tailored Algorithms for Anomaly Detection in Photovoltaic Systems,” *Energies*, vol. 13, no. 1, p. 225, Jan. 2020. [Online]. Available: <https://www.mdpi.com/1996-1073/13/1/225>
- [21] G. M. Tina, C. Ventura, S. Ferlito, and S. De Vito, “A State-of-Art-Review on Machine-Learning Based Methods for PV,” *Applied Sciences*, vol. 11, no. 16, p. 7550, Aug. 2021. [Online]. Available: <https://www.mdpi.com/2076-3417/11/16/7550>
- [22] J. Schmidhuber, “Deep learning in neural networks: An overview,” *Neural Networks*, vol. 61, pp. 85–117, Jan. 2015. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S0893608014002135>

- [23] I. Sutskever, *Training recurrent neural networks*. University of Toronto Toronto, ON, Canada, 2013.
- [24] L. Jiao, R. Shang, F. Liu, and W. Zhang, “Chapter 2 - the models and structure of neural networks,” in *Brain and Nature-Inspired Learning Computation and Recognition*, L. Jiao, R. Shang, F. Liu, and W. Zhang, Eds. Elsevier, 2020, pp. 47–79. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/B9780128197950000025>
- [25] J. Terven, D.-M. Cordova-Esparza, A. Ramirez-Pedraza, and E. Chávez Urbiola, “Loss functions and metrics in deep learning. a review,” 07 2023.
- [26] A. Sherstinsky, “Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network,” *Physica D: Nonlinear Phenomena*, vol. 404, p. 132306, Mar. 2020. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S0167278919305974>
- [27] S. Hochreiter, “Untersuchungen zu dynamischen neuronalen netzen,” *Diploma, Technische Universität München*, vol. 91, no. 1, p. 31, 1991.
- [28] V. N. A. Vapnik, *The nature of statistical learning theory*. New York : Springer, 2010. [Online]. Available: <http://archive.org/details/natureofstatisti0000vapn>
- [29] A. J. Smola and B. Schölkopf, “A tutorial on support vector regression,” *Statistics and Computing*, vol. 14, no. 3, pp. 199–222, Aug. 2004. [Online]. Available: <http://link.springer.com/10.1023/B:STCO.0000035301.49549.88>
- [30] L. Deng, “Three Classes of Deep Learning Architectures and Their Applications: A Tutorial Survey,” 2012.
- [31] F. Harrou, A. Dairi, B. Taghezouit, and Y. Sun, “An unsupervised monitoring procedure for detecting anomalies in photovoltaic systems using a one-class Support Vector Machine,” *Solar Energy*, vol. 179, pp. 48–58, Feb. 2019. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S0038092X18312209>
- [32] L. Longi Green Energy Technology Co., “Lgile-t-tmd-059-108 lr5-72hbd 530-550m (35-3015 frame dual certification) v14,” shaanxi, China. Accessed July 24, 2024. [Online]. Available: https://static.longi.com/L_Gi_LE_T_TMD_059_108_LR_5_72_HBD_530_550_M_35_30_and_15_V14_4c79e9b9a7.pdf
- [33] G. Xydis, “The wind chill temperature effect on a large-scale PV plant—an exergy approach,” *Progress in Photovoltaics: Research and Applications*, vol. 21, no. 8, pp. 1611–1624, Dec. 2013. [Online]. Available: <https://onlinelibrary.wiley.com/doi/10.1002/pip.2247>
- [34] A. K. Jain, “Data clustering: 50 years beyond k-means,” *Pattern Recognition Letters*, vol. 31, no. 8, pp. 651–666, 2010, award winning papers from the 19th International Conference on Pattern Recognition (ICPR). [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0167865509002323>
- [35] P. Drineas, A. Frieze, R. Kannan, S. Vempala, and V. Vinay, “Clustering Large Graphs via the Singular Value Decomposition,” *Machine Learning*, vol. 56, no. 1-3, pp. 9–33, Jul. 2004. [Online]. Available: <http://link.springer.com/10.1023/B:MACH.0000033113.59016.96>
- [36] A. K. Jain and R. C. Dubes, *Algorithms for clustering data*, ser. Prentice Hall advanced reference series. Englewood Cliffs, N.J: Prentice Hall, 1988.

- [37] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 11 1997. [Online]. Available: <https://doi.org/10.1162/neco.1997.9.8.1735>
- [38] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. VanderPlas, A. Joly, B. Holt, and G. Varoquaux, “API design for machine learning software: experiences from the scikit-learn project,” in *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, 2013, pp. 108–122.
- [39] H. W. Kuhn, “The Hungarian Method for the Assignment Problem,” in *50 Years of Integer Programming 1958-2008*, M. Jünger, T. M. Liebling, D. Naddef, G. L. Nemhauser, W. R. Pulleyblank, G. Reinelt, G. Rinaldi, and L. A. Wolsey, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 29–47. [Online]. Available: http://link.springer.com/10.1007/978-3-540-68279-0_2

Appendix A

A.1 K-Means Model Runs

Each of the 20 following images represents a run of the K-Means-based algorithm. The algorithm was used 20 times on the data for string 15 in January 2024. Each image is accompanied by a table with the respective values.

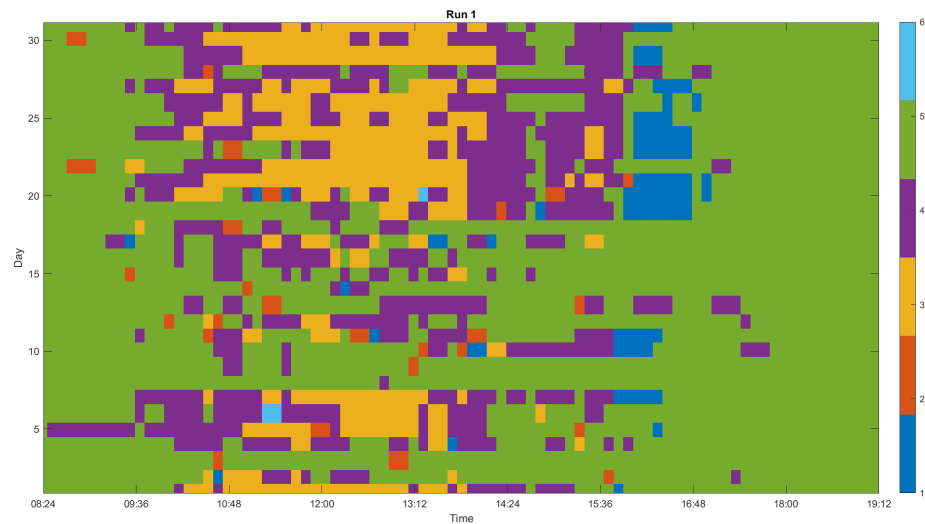


Figure A.1: Run 1

Table A.1: Total power and timestamp count per cluster for string 15 in January - Run 1.

Cluster	Total Power in kW	Timestamp Count	Total Power in kW*	Timestamp Count *
1	5.9	61	1.0	50
2	8.0	27	1.4	5
3	4.2	266	0.7	8
4	7.5	417	2.3	91
5	15.1	1583	3.2	435
6	5.1	2	0.0	0

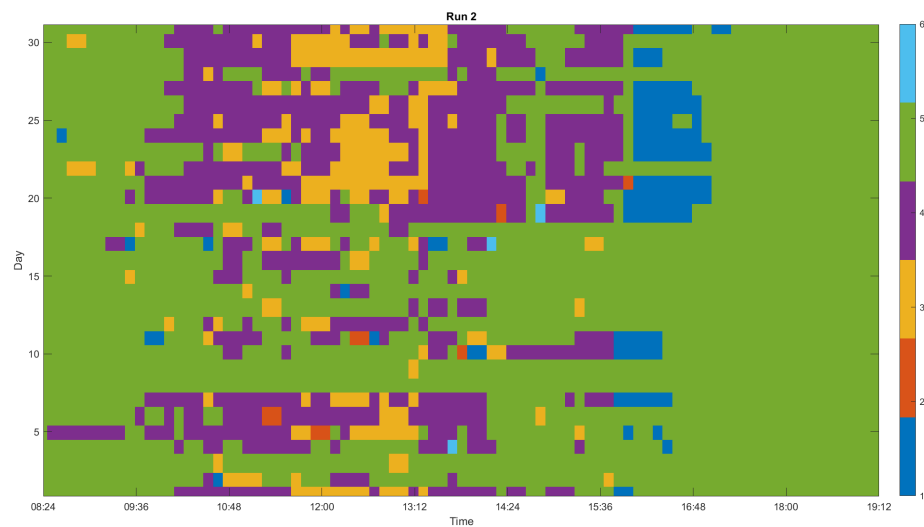


Figure A.2: Run 2

Table A.2: Total power and timestamp count per cluster for string 15 in January - Run 2.

Cluster	Total Power in kW	Timestamp Count	Total Power in kW*	Timestamp Count *
1	6.6	78	1.1	67
2	5.7	7	0.0	1
3	4.3	151	0.9	5
4	6.6	433	1.0	68
5	14.8	1683	3.3	448
6	5.8	4	0.0	0

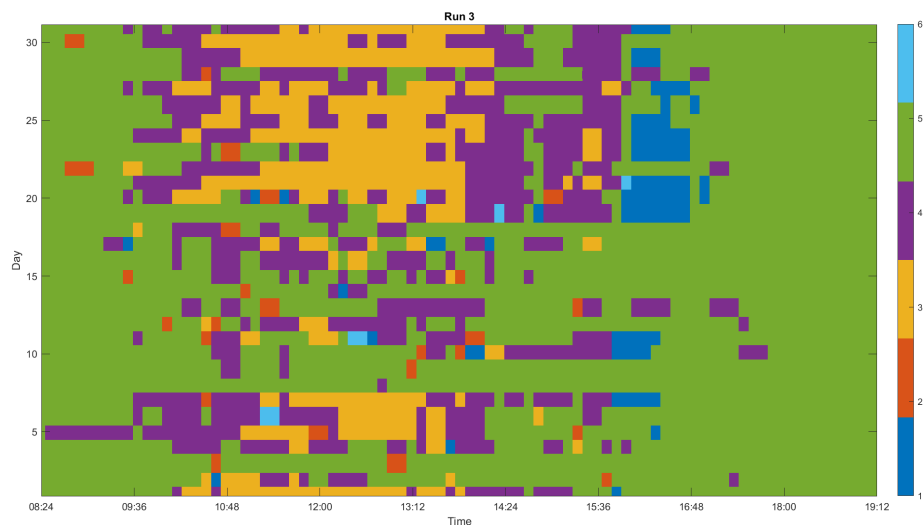


Figure A.3: Run 3

Table A.3: Total power and timestamp count per cluster for string 15 in January - Run 3.

Cluster	Total Power in kW	Timestamp Count	Total Power in kW*	Timestamp Count *
1	5.9	61	1.0	50
2	7.7	26	1.0	4
3	4.2	265	0.7	8
4	7505.359	417	2.3	91
5	15.1	1582	3.2	435
6	6.4	5	0.0	1

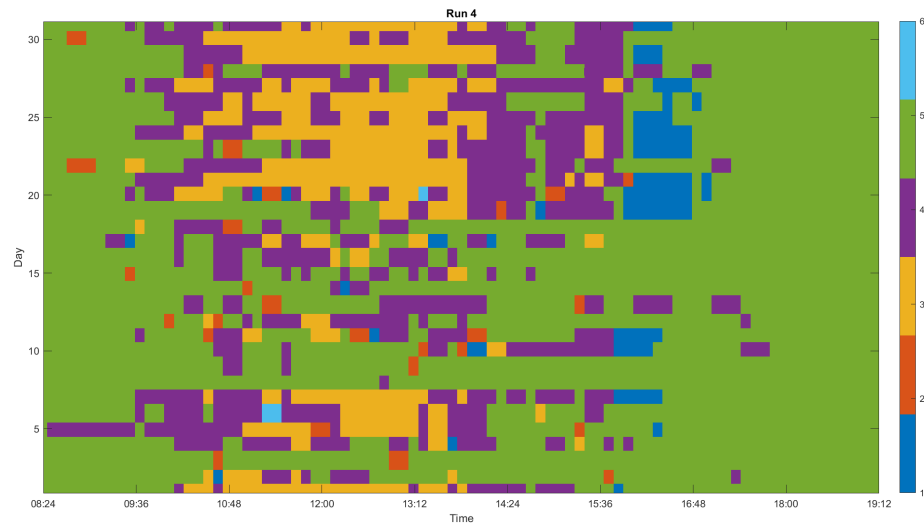


Figure A.4: Run 4

Table A.4: Total power and timestamp count per cluster for string 15 in January - Run 4.

Cluster	Total_Power	Timestamp_Count	Total_Power_*	Timestamp_Count_*
1	5.9	61	1.0	50
2	8.0	27	1.4	5
3	4.2	266	7.2	8
4	7.5	417	2.3	91
5	15.1	1583	3.2	435
6	5.1	2	0.0	0

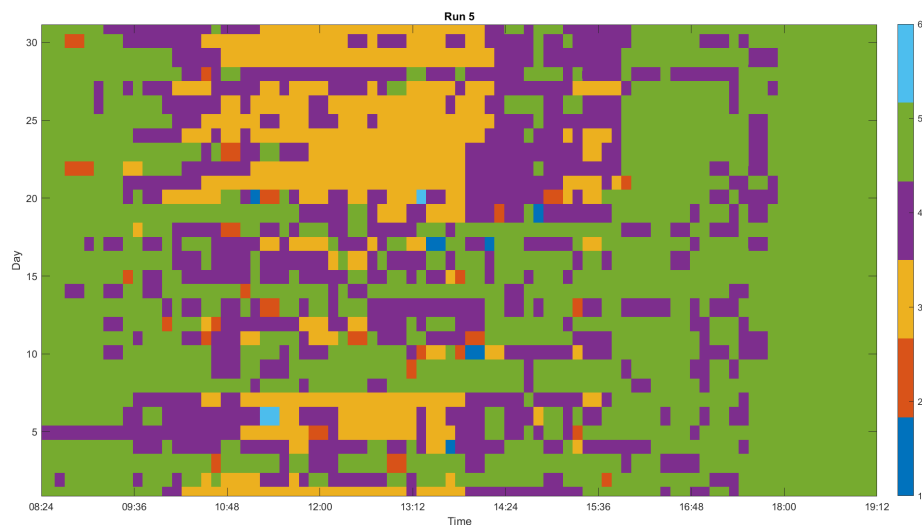


Figure A.5: Run 5

Table A.5: Total power and timestamp count per cluster for string 15 in January - Run 5.

Cluster	Total Power in kW	Timestamp Count	Total Power in kW*	Timestamp Count *
1.0	4.5	6	0.0	0
2.0	8.0	28	1.3	5
3.0	4.6	309	0.7	18
4.0	8.9	559	3.4	168
5.0	15.4	1452	3.1	398
6.0	5.1	2	0.0	0

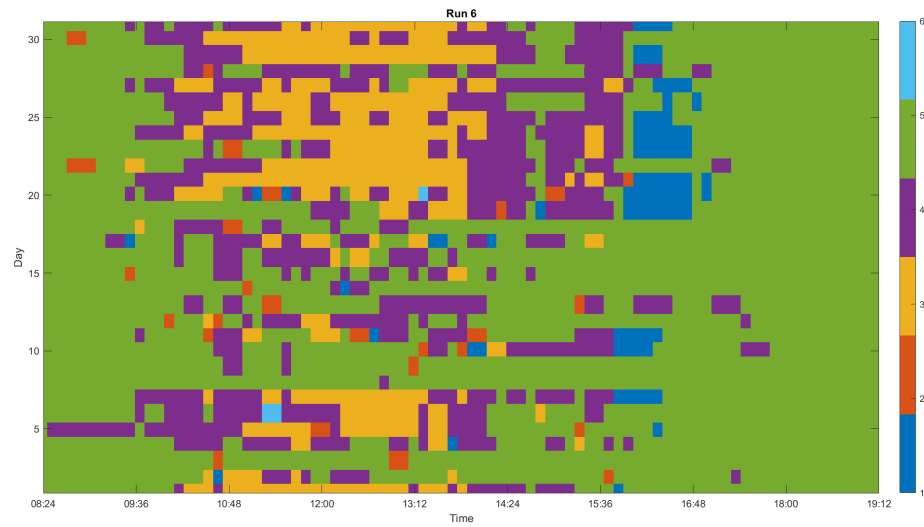


Figure A.6: Run 6

Table A.6: Total power and timestamp count per cluster for string 15 in January - Run 6.

Cluster	Total Power in kW	Timestamp Count	Total Power in kW*	Timestamp Count *
1.0	5.9	61	1.0	50
2.0	8.0	27	1.4	5
3.0	4.2	266	0.7	8
4.0	7.5	417	2.3	91
5.0	15.1	1583	3.2	435
6.0	5.1	2	0.0	0

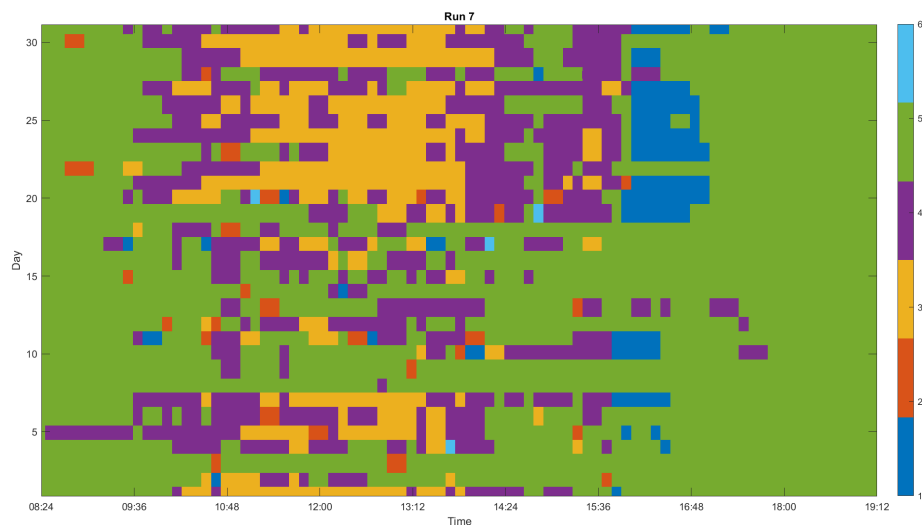


Figure A.7: Run 7

Table A.7: Total power and timestamp count per cluster for string 15 in January - Run 7.

Cluster	Total Power in kW	Timestamp Count	Total Power in kW*	Timestamp Count *
1.0	6.1	76	1.1	66
2.0	7.9	28	1.4	5
3.0	4.2	262	0.7	8
4.0	7.3	407	2.1	83
5.0	15.1	1579	3.3	427
6.0	5.8	4	0.0	0

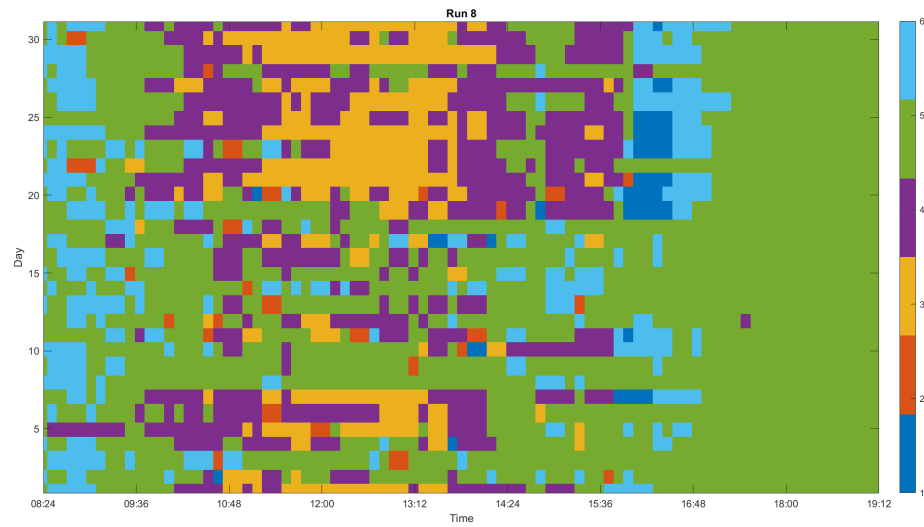


Figure A.8: Run 8

Table A.8: Total power and timestamp count per cluster for string 15 in January - Run 8.

Cluster	Total Power in kW	Timestamp Count	Total Power in kW*	Timestamp Count *
1.0	5.3	33	0.7	26
2.0	7.7	27	1.5	4
3.0	4.0	228	451.664	6
4.0	7.0	380	1.3	71
5.0	14.7	1440	3.2	393
6.0	12.6	248	2.2	89

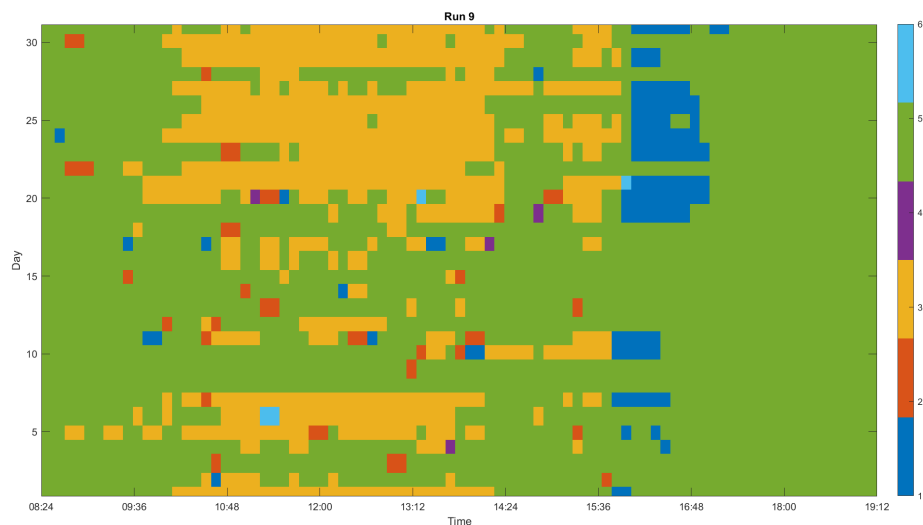


Figure A.9: Run 9

Table A.9: Total power and timestamp count per cluster for string 15 in January - Run 9.

Cluster	Total Power in kW	Timestamp Count	Total Power in kW*	Timestamp Count *
1.0	6.6	78	1.1	67
2.0	7.6	28	9.8	4
3.0	5.6	460	8.5	48
4.0	5.8	4	0.0	0
5.0	14.5	1783	3.4	469
6.0	8.4	3	0.0	1

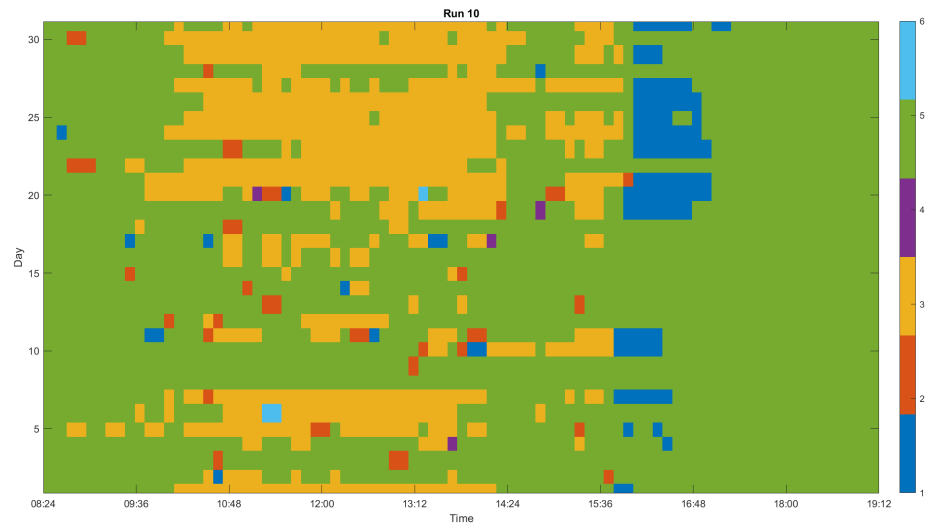


Figure A.10: Run 10

Table A.10: Total power and timestamp count per cluster for string 15 in January - Run 10.

Cluster	Total Power in kW	Timestamp Count	Total Power in kW*	Timestamp Count *
1.0	6.6	78	1.1	67
2.0	7.9	29	1.4	5
3.0	5.6	460	0.8	48
4.0	5.8	4	0.0	0
5.0	14.5	1783	3.4	469
6.0	5.1	2	0.0	0

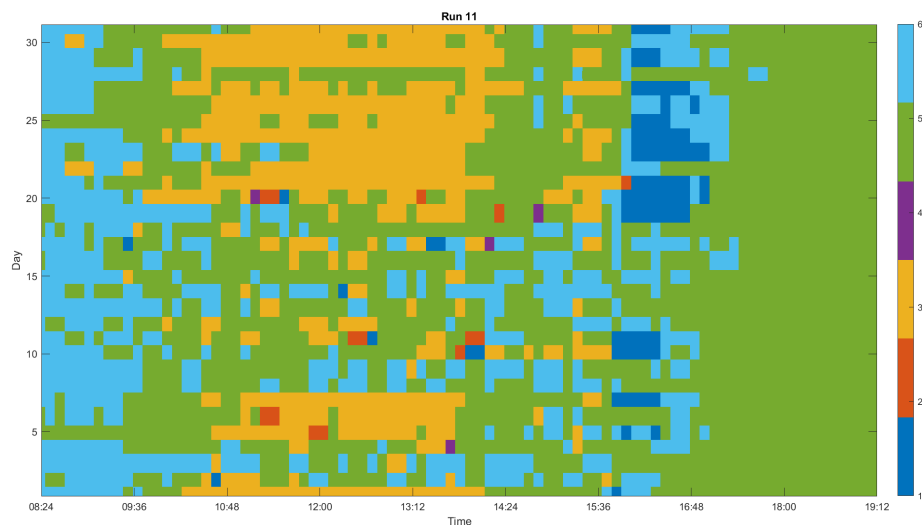


Figure A.11: Run 11

Table A.11: Total power and timestamp count per cluster for string 15 in January - Run 11.

Cluster	Total_Power	Timestamp_Count	Total_Power_*	Timestamp_Count_*
1.0	5.4	62	1.1	55
2.0	5.6	9	0.0	1
3.0	5.3	394	0.7	36
4.0	5.8	4	0.0	0
5.0	13.9	1404	3.5	375
6.0	11.6	483	2.5	122

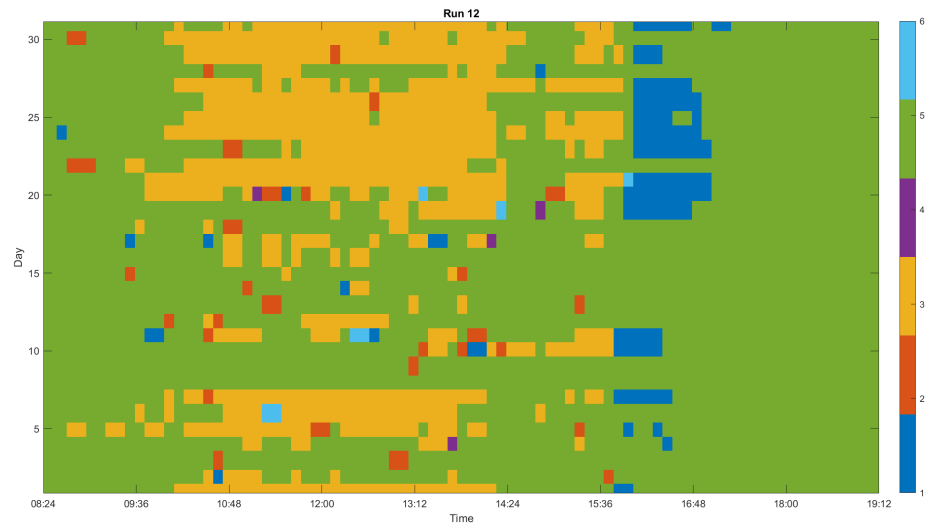


Figure A.12: Run 12

Table A.12: Total power and timestamp count per cluster for string 15 in January - Run 12.

Cluster	Total Power in kW	Timestamp Count	Total Power in kW*	Timestamp Count *
1.0	6.6	78	1.1	67
2.0	7313.728	30	1.0	4
3.0	5.6	458	0.8	49
4.0	5.8	4	0.0	0
5.0	14.5	1781	3.4	468
6.0	6.4	5	0.0	1

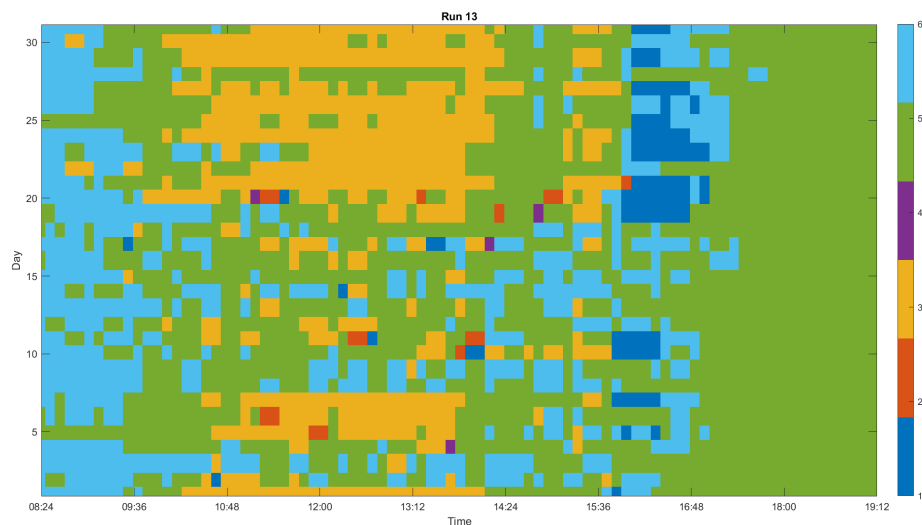


Figure A.13: Run 13

Table A.13: Total power and timestamp count per cluster for string 15 in January - Run 13.

Cluster	Total Power in kW	Timestamp Count	Total Power in kW*	Timestamp Count *
1.0	5.4	62	1.1	55
2.0	5.6	10	2.5	2
3.0	5.4	397	7.1	36
4.0	5.8	4	0.0	0
5.0	13.9	1401	3.5	374
6.0	11.6	482	2.5	122

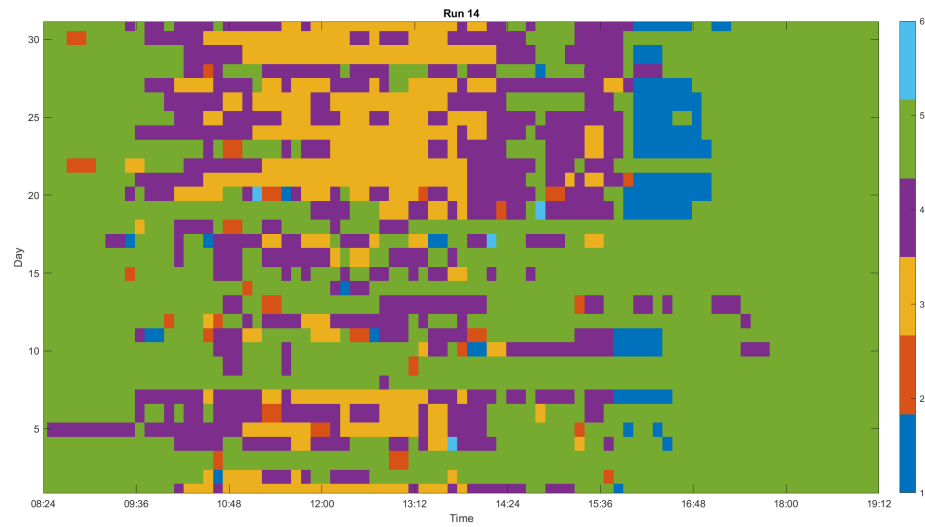


Figure A.14: Run 14

Table A.14: Total power and timestamp count per cluster for string 15 in January - Run 14.

Cluster	Total Power in kW	Timestamp Count	Total Power in kW*	Timestamp Count *
1.0	6.0	76	1.1	66
2.0	7.9	28	1.4	5
3.0	4.2	262	0.7	8
4.0	7.3	407	2.1	83
5.0	15.1	1579	3.3	427
6.0	5.8	4	0.0	0

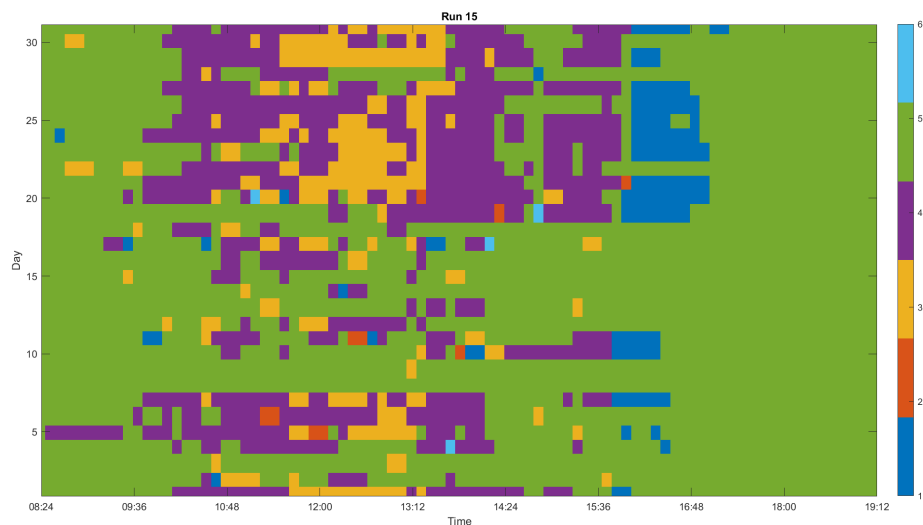


Figure A.15: Run 15

Table A.15: Total power and timestamp count per cluster for string 15 in January - Run 15.

Cluster	Total Power in kW	Timestamp Count	Total Power in kW*	Timestamp Count *
1.0	6.6	78	1.1	67
2.0	5.7	7	0.0	1
3.0	4.4	162	0.9	5
4.0	6.6	429	1.0	68
5.0	14.8	1676	3.3	448
6.0	5.8	4	0.0	0

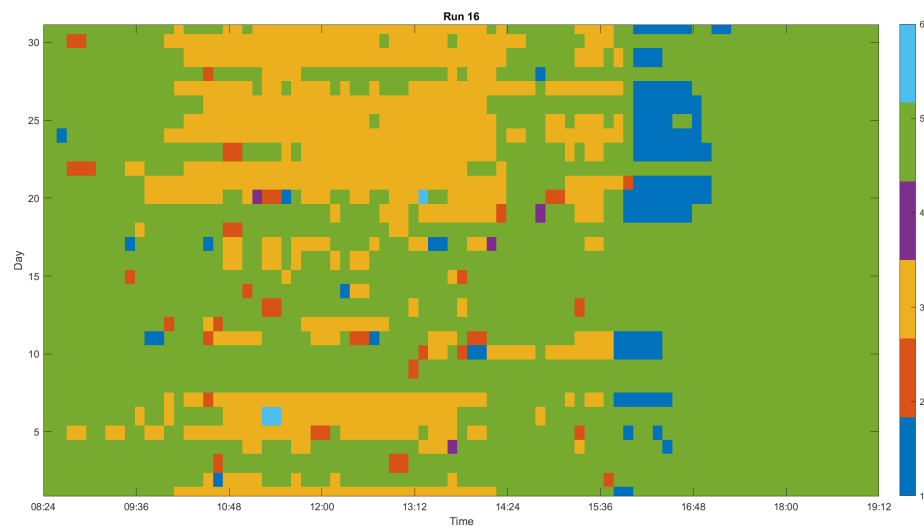


Figure A.16: Run 16

Table A.16: Total power and timestamp count per cluster for string 15 in January - Run 16.

Cluster	Total Power in kW	Timestamp Count	Total Power in kW*	Timestamp Count *
1.0	6.6	78	1.1	67
2.0	7.9	29	1.4	5
3.0	5.6	460	0.8	48
4.0	5.8	4	0.0	0
5.0	14.5	1783	3.4	469
6.0	5.1	2	0.0	0

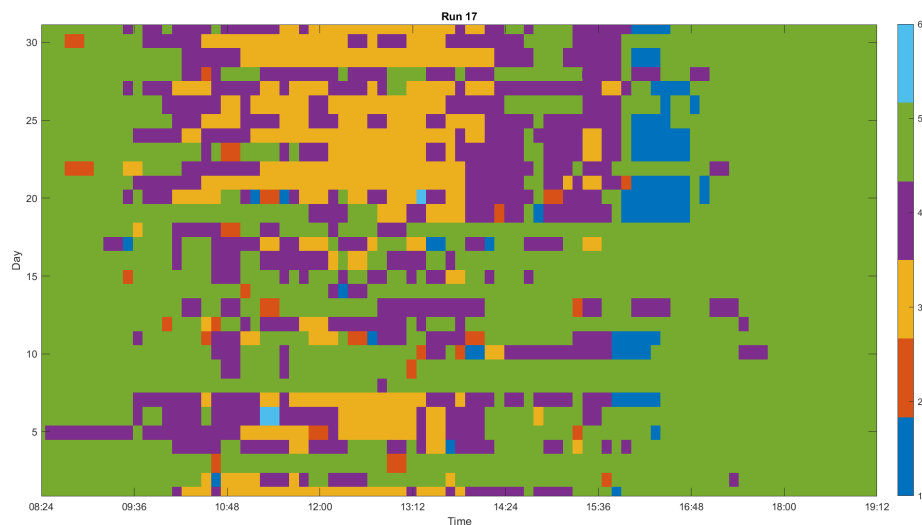


Figure A.17: Run 17

Table A.17: Total power and timestamp count per cluster for string 15 in January - Run 17.

Cluster	Total Power in kW	Timestamp Count	Total Power in kW*	Timestamp Count *
1.0	5.9	61	1.0	50
2.0	8.0	27	1.4	5
3.0	4.2	266	0.7	8
4.0	7.5	417	2.3	91
5.0	15.1	1583	3.2	435
6.0	5.1	2	0.0	0

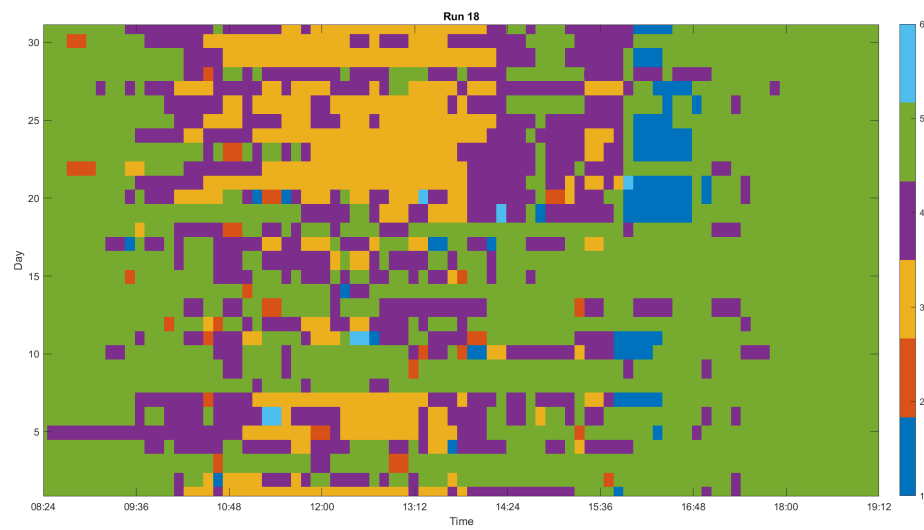


Figure A.18: Run 18

Table A.18: Total power and timestamp count per cluster for string 15 in January - Run 18.

Cluster	Total Power in kW	Timestamp Count	Total Power in kW*	Timestamp Count *
1.0	5.9	61	1.0	50
2.0	7.7	26	1.0	4
3.0	4.5	284	0.7	14
4.0	8.0	450	2.8	106
5.0	15.2	1530	3.2	414
6.0	6.4	5	0.0	1

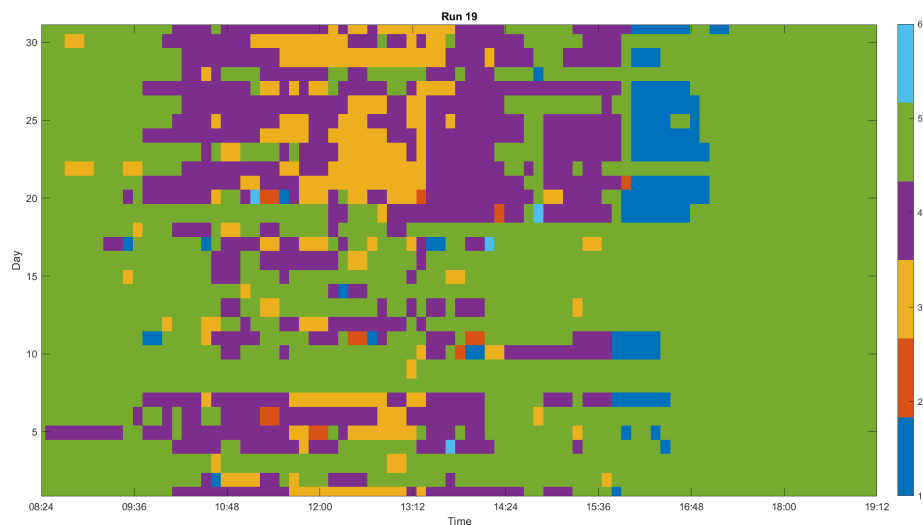


Figure A.19: Run 19

Table A.19: Total power and timestamp count per cluster for string 15 in January - Run 19.

Cluster	Total Power in kW	Timestamp Count	Total Power in kW*	Timestamp Count *
1.0	6.0	76	1.1	66
2.0	5.6	9	0.0	1
3.0	4.3	170	0.9	5
4.0	6.7	436	1.0	70
5.0	14.8	1661	3.3	447
6.0	5.8	4	0.0	0

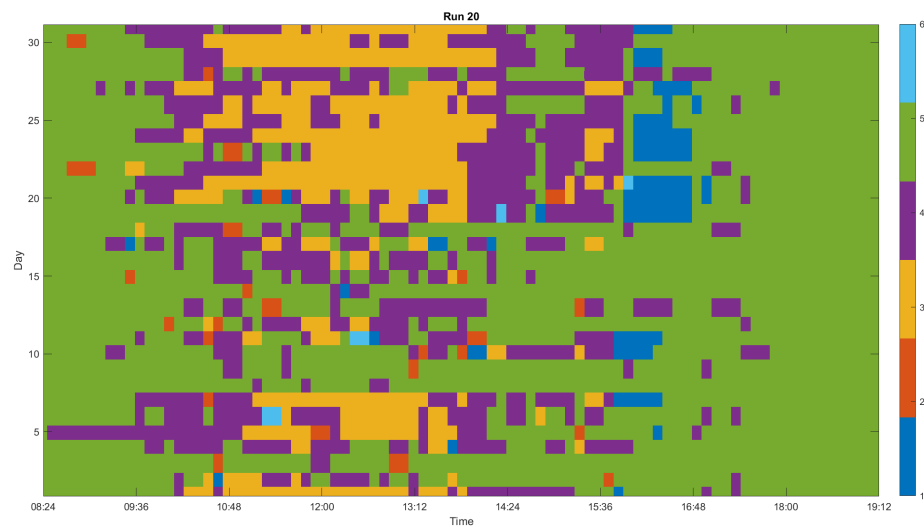


Figure A.20: Run 20

Table A.20: Total power and timestamp count per cluster for string 15 in January - Run 20.

Cluster	Total Power in kW	Timestamp Count	Total Power in kW*	Timestamp Count *
1.0	5.9	61	1.0	50
2.0	7.7	26	1.0	4
3.0	4.5	284	0.7	14
4.0	7.9	450	2.8	106
5.0	15.2	1530	3.2	414
6.0	6.4	5	0.0	1

A.2 LSTM Models

Each of the 20 following images represents a model with the discussed LSTM architecture. The string 15 data in January 2024 was fed to these models. Each image is accompanied by a table with the respective values.

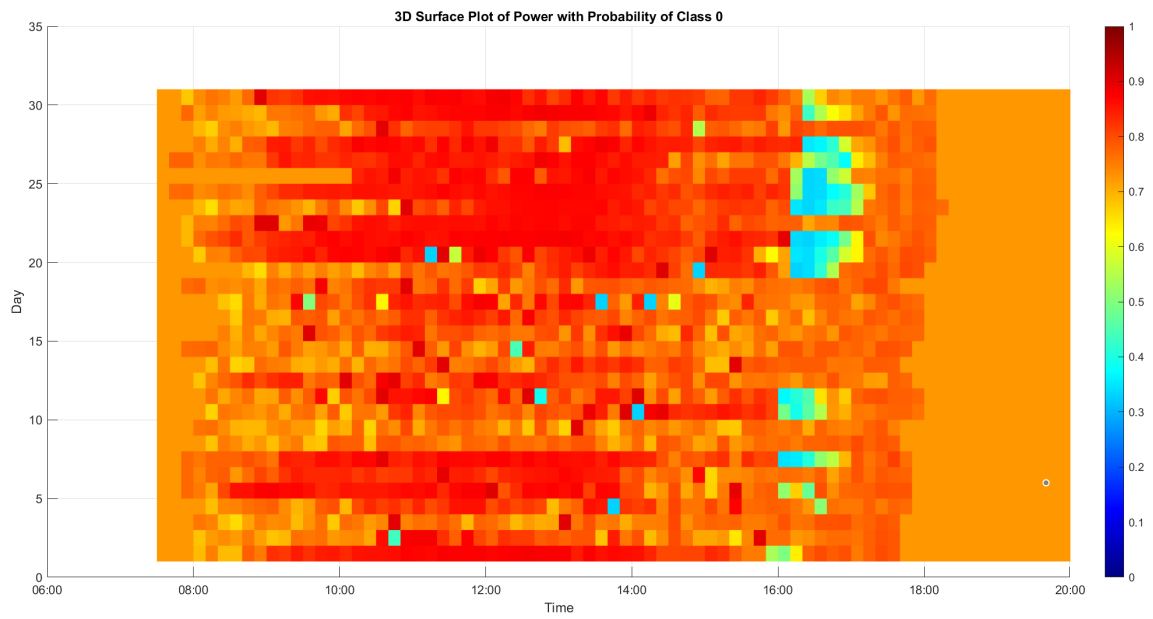


Figure A.21: Model 1

Table A.21: Total power per probability of Class 0 interval for string 15 in January - Model 1

Probability Interval	Sum of Standard Error in kW	Timestamp Count	Sum of Standard Error in kW *	Timestamp Count *
0.0-0.1	0.0	0	0.0	0
0.1-0.2	0.0	0	0.0	0
0.2-0.3	0.0	0	0.0	0
0.3-0.4	49.3	33	28.7	26
0.4-0.5	13.0	20	11.7	18
0.5-0.6	9.2	22	7.6	18
0.6-0.7	12.6	104	4.5	31
0.7-0.8	-116.8	1411	-50.5	374
0.8-0.9	-507.1	746	-46.8	88
0.9-1.0	-51.0	20	-6.7	3

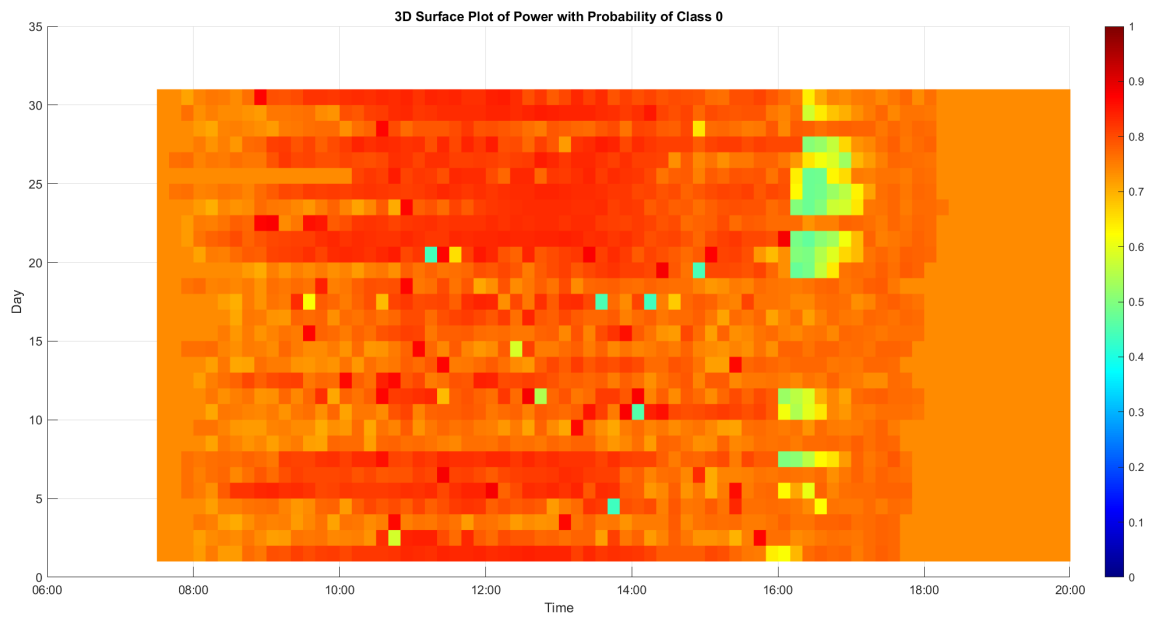


Figure A.22: Model 2

Table A.22: Total power per probability of Class 0 interval for string 15 in January - Model 2

Probability Interval	Sum of Standard Error in kW	Timestamp Count	Sum of Standard Error in kW *	Timestamp Count *
0.0-0.1	0.0	0	0.0	0
0.1-0.2	0.0	0	0.0	0
0.2-0.3	0.0	0	0.0	0
0.3-0.4	0.0	0	0.0	0
0.4-0.5	33.9	17	14.1	11
0.5-0.6	25.7	31	23.5	28
0.6-0.7	14.2	37	12.1	31
0.7-0.8	-19.3	1735	-66.5	447
0.8-0.9	-471.8	536	-34.8	41
0.9-1.0	0.0	0	0.0	0

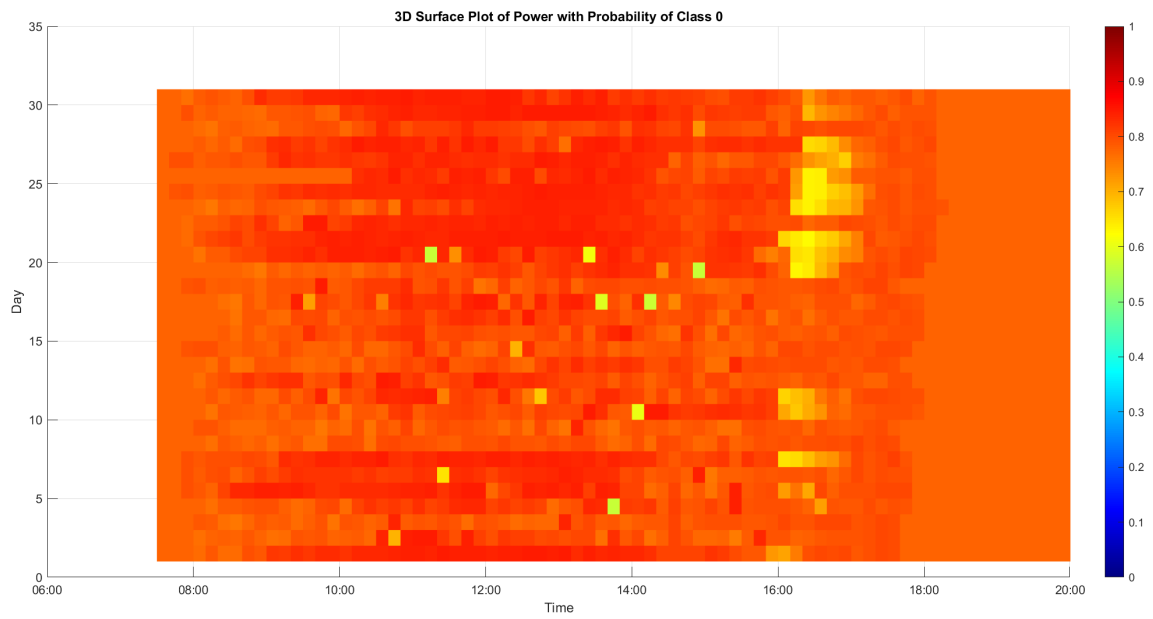


Figure A.23: Model 3

Table A.23: Total power per probability of Class 0 interval for string 15 in January - Model 3

Probability Interval	Sum of Standard Error in kW	Timestamp Count	Sum of Standard Error in kW *	Timestamp Count *
0.0-0.1	0.0	0	0.0	0
0.1-0.2	0.0	0	0.0	0
0.2-0.3	0.0	0	0.0	0
0.3-0.4	0.0	0	0.0	0
0.4-0.5	0.0	0	0.0	0
0.5-0.6	17.8	5	0.0	0
0.6-0.7	21.6	43	30.3	37
0.7-0.8	-67.7	1405	-19.2	373
0.8-0.9	-562.6	903	-62.8	148
0.9-1.0	0.0	0	0.0	0

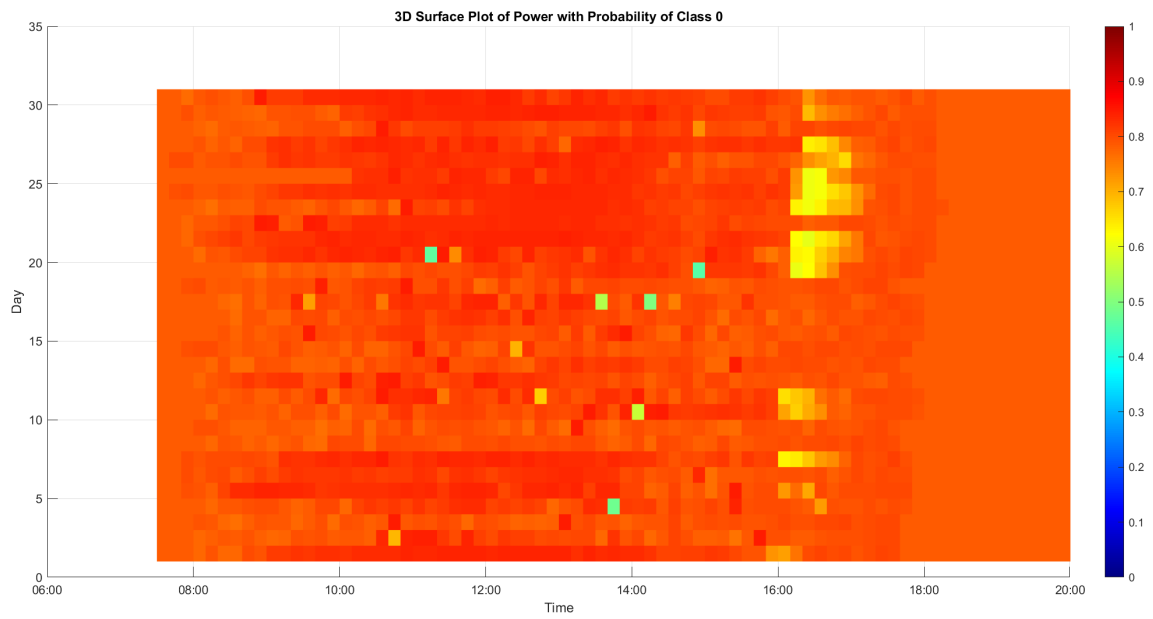


Figure A.24: Model 4

Table A.24: Total power per probability of Class 0 interval for string 15 in January - Model 4

Probability Interval	Sum of Standard Error in kW	Timestamp Count	Sum of Standard Error in kW *	Timestamp Count *
0.0-0.1	0.0	0	0.0	0
0.1-0.2	0.0	0	0.0	0
0.2-0.3	0.0	0	0.0	0
0.3-0.4	0.0	0	0.0	0
0.4-0.5	15.6	4	0.0	0
0.5-0.6	4.1	2	0.0	0
0.6-0.7	38.7	40	36.5	37
0.7-0.8	-46.8	1335	-12.3	337
0.8-0.9	-60.2	975	-75.6	184
0.9-1.0	0.0	0	0.0	0

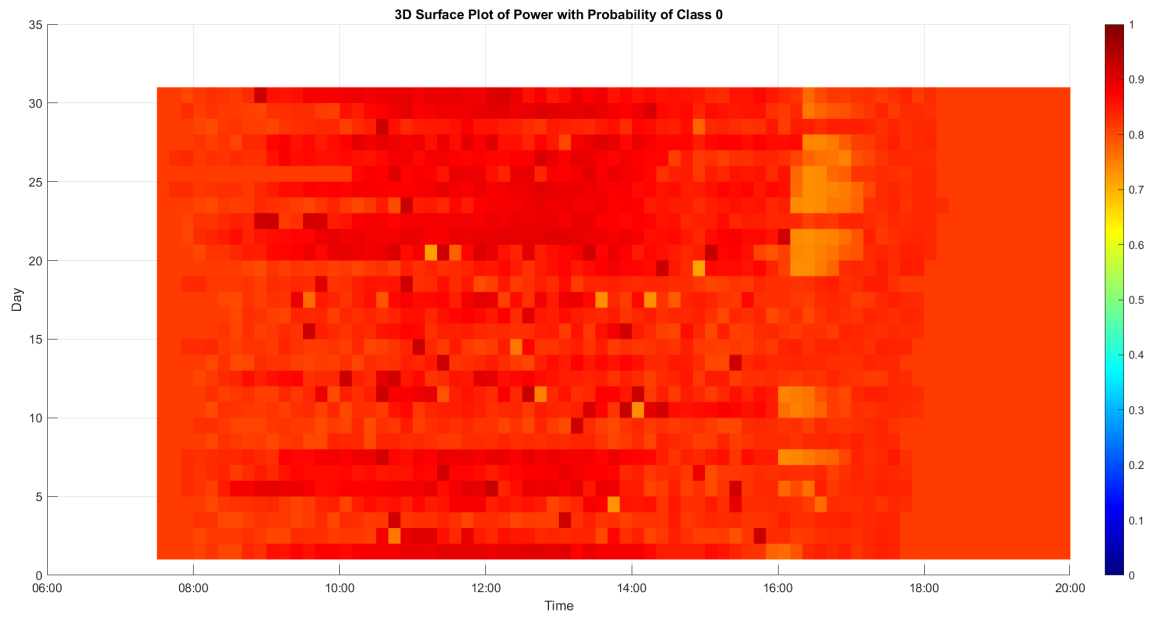


Figure A.25: Model 5

Table A.25: Total power per probability of Class 0 interval for string 15 in January - Model 5

Probability Interval	Sum of Standard Error in kW	Timestamp Count	Sum of Standard Error in kW *	Timestamp Count *
0.0-0.1	0.0	0	0.0	0
0.1-0.2	0.0	0	0.0	0
0.2-0.3	0.0	0	0.0	0
0.3-0.4	0.0	0	0.0	0
0.4-0.5	0.0	0	0.0	0
0.5-0.6	0.0	0	0.0	0
0.6-0.7	0.0	0	0.0	0
0.7-0.8	76.0	99	50.5	74
0.8-0.9	-553.3	2202	-88.0	479
0.9-1.0	-113.5	55	-14.0	5

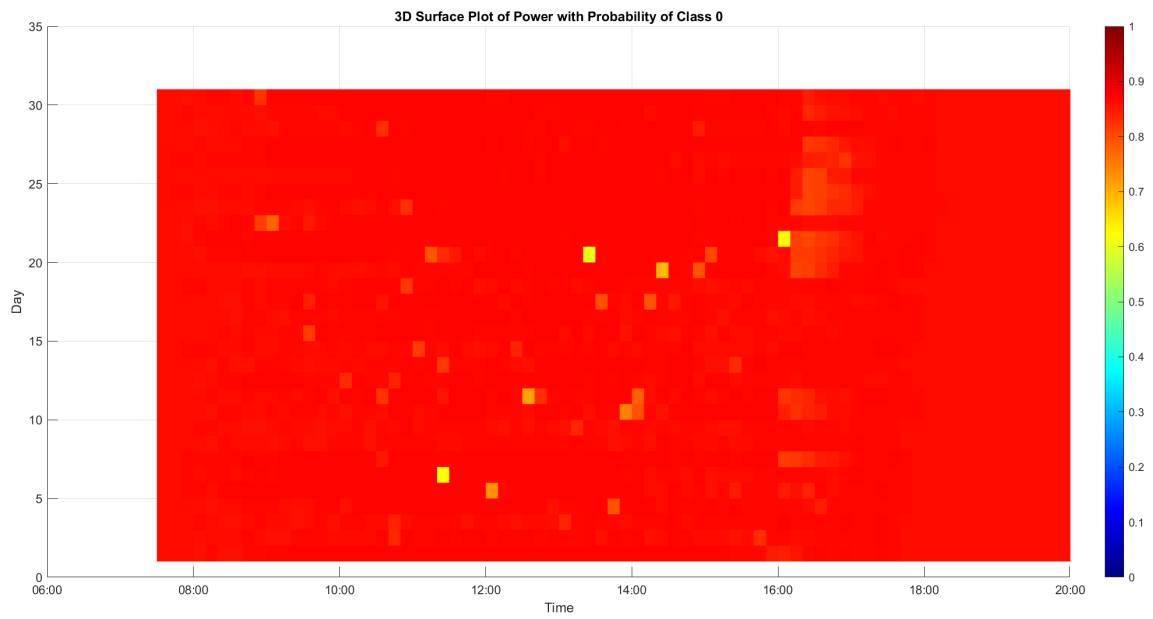


Figure A.26: Model 6

Table A.26: Total power per probability of Class 0 interval for string 15 in January - Model 6

Probability Interval	Sum of Standard Error in kW	Timestamp Count	Sum of Standard Error in kW *	Timestamp Count *
0.0-0.1	0.0	0	0.0	0
0.1-0.2	0.0	0	0.0	0
0.2-0.3	0.0	0	0.0	0
0.3-0.4	0.0	0	0.0	0
0.4-0.5	0.0	0	0.0	0
0.5-0.6	0.0	0	0.0	0
0.6-0.7	-22.3	4	-5.5	1
0.7-0.8	3.8	11	0.0	0
0.8-0.9	-572.3	2341	-46.0	557
0.9-1.0	0.0	0	0.0	0

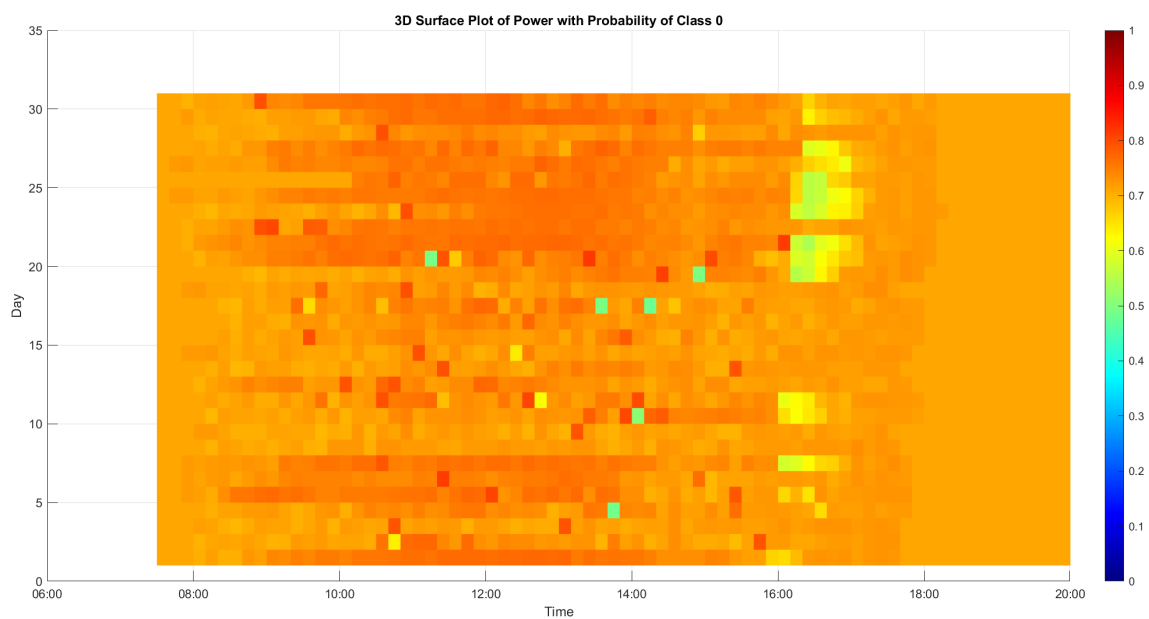


Figure A.27: Model 7

Table A.27: Total power per probability of Class 0 interval for string 15 in January - Model 7

Probability Interval	Sum of Standard Error in kW	Timestamp Count	Sum of Standard Error in kW *	Timestamp Count *
0.0-0.1	0.0	0	0.0	0
0.1-0.2	0.0	0	0.0	0
0.2-0.3	0.0	0	0.0	0
0.3-0.4	0.0	0	0.0	0
0.4-0.5	17.8	5	0.0	0
0.5-0.6	22.5	18	20.6	17
0.6-0.7	40.2	113	31.4	69
0.7-0.8	-633.1	2211	-98.0	471
0.8-0.9	-38.3	9	-5.5	1
0.9-1.0	0.0	0	0.0	0

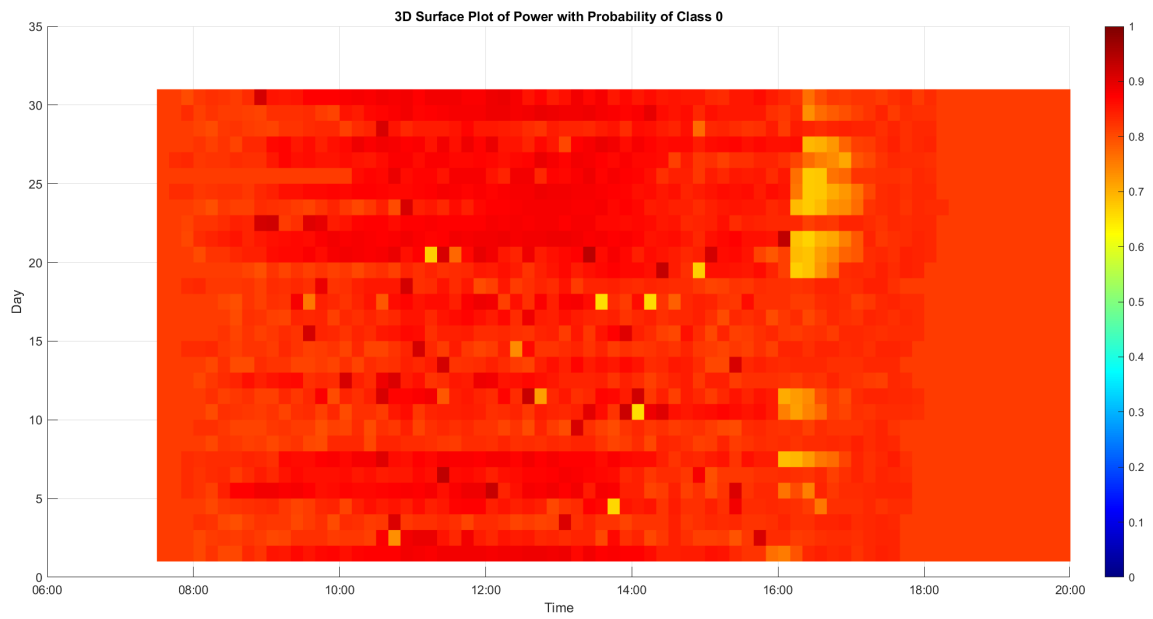


Figure A.28: Model 8

Table A.28: Total power per probability of Class 0 interval for string 15 in January - Model 8

Probability Interval	Sum of Standard Error in kW	Timestamp Count	Sum of Standard Error in kW *	Timestamp Count *
0.0-0.1	0.0	0	0.0	0
0.1-0.2	0.0	0	0.0	0
0.2-0.3	0.0	0	0.0	0
0.3-0.4	0.0	0	0.0	0
0.4-0.5	0.0	0	0.0	0
0.5-0.6	0.0	0	0.0	0
0.6-0.7	40.3	23	20.6	17
0.7-0.8	37.4	88	30.5	61
0.8-0.9	-583.2	2213	-88.5	475
0.9-1.0	-85.3	32	-14.0	5

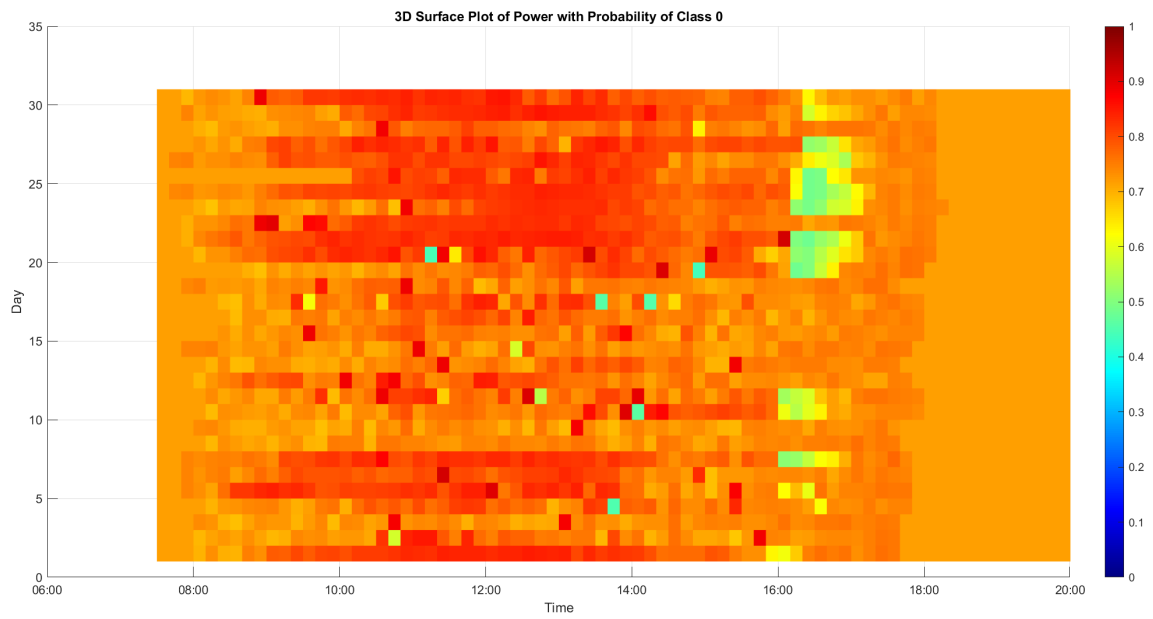


Figure A.29: Model 9

Table A.29: Total power per probability of Class 0 interval for string 15 in January - Model 9

Probability Interval	Sum of Standard Error in kW	Timestamp Count	Sum of Standard Error in kW *	Timestamp Count *
0.0-0.1	0.0	0	0.0	0
0.1-0.2	0.0	0	0.0	0
0.2-0.3	0.0	0	0.0	0
0.3-0.4	0.0	0	0.0	0
0.4-0.5	30.3	14	10.6	8
0.5-0.6	292.2	34	27.1	31
0.6-0.7	22.3	102	14.4	49
0.7-0.8	-259.9	1781	-81.5	453
0.8-0.9	-380.1	418	-16.7	16
0.9-1.0	-32.7	7	-5.5	1

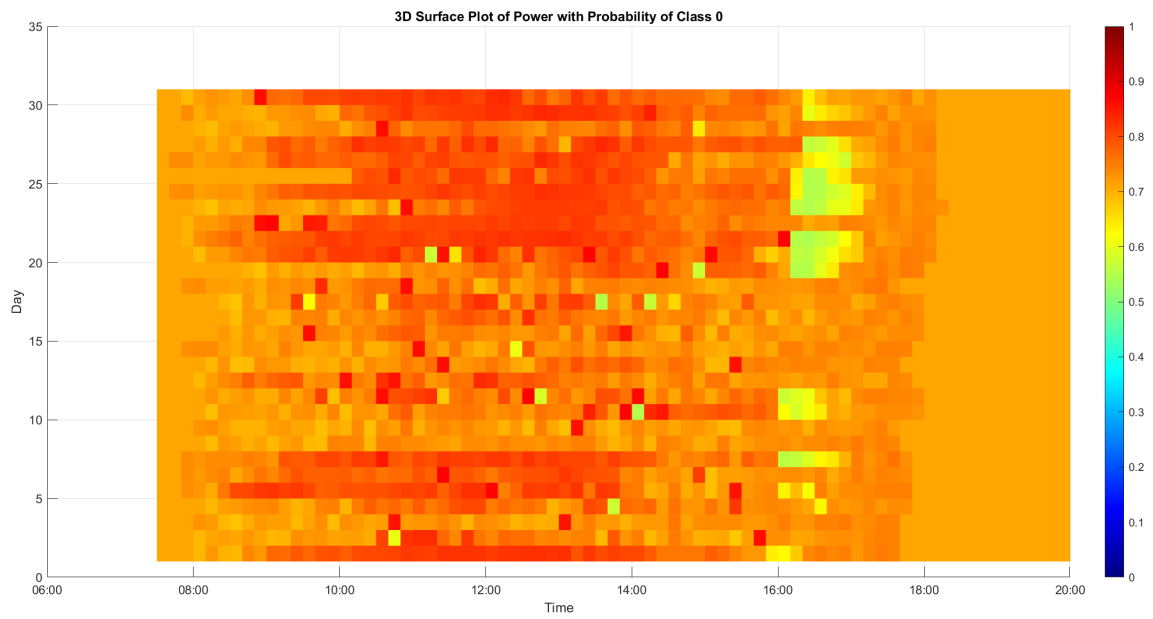


Figure A.30: Model 10

Table A.30: Total power per probability of Class 0 interval for string 15 in January - Model 10

Probability Interval	Sum of Standard Error in kW	Timestamp Count	Sum of Standard Error in kW *	Timestamp Count *
0.0-0.1	0.0	0	0.0	0
0.1-0.2	0.0	0	0.0	0
0.2-0.3	0.0	0	0.0	0
0.3-0.4	0.0	0	0.0	0
0.4-0.5	0.0	0	0.0	0
0.5-0.6	53.7	39	33.2	32
0.6-0.7	30.5	142	19.4	62
0.7-0.8	-341.5	1873	-87.6	456
0.8-0.9	-333.6	302	-16.5	8
0.9-1.0	0.0	0	0.0	0

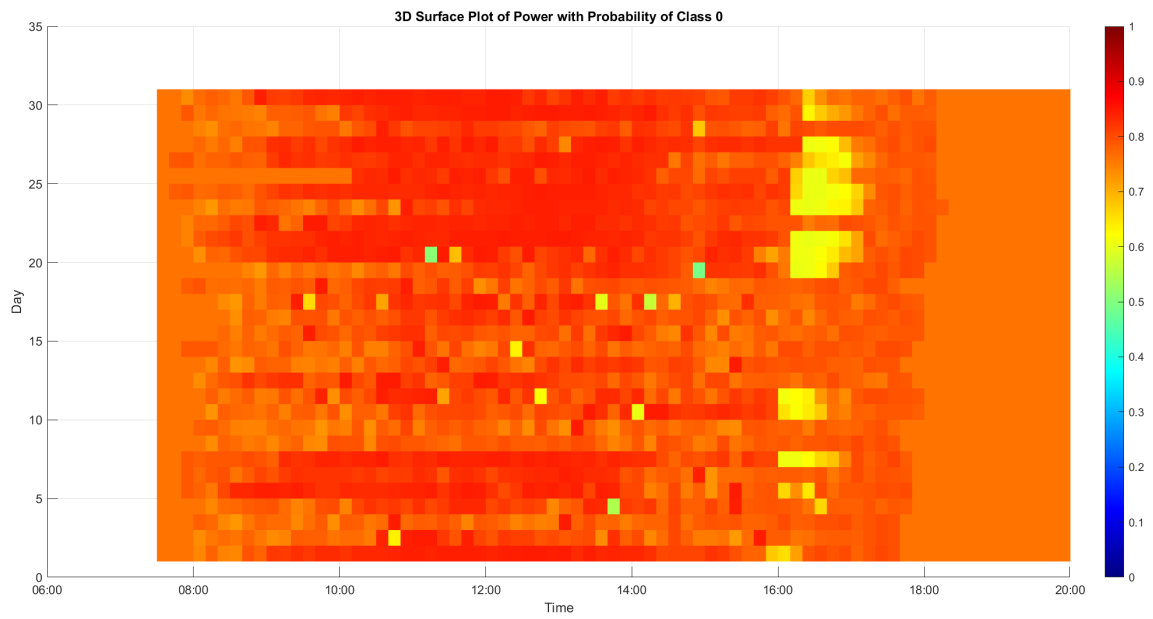


Figure A.31: Model 11

Table A.31: Total power per probability of Class 0 interval for string 15 in January - Model 11

Probability Interval	Sum of Standard Error in kW	Timestamp Count	Sum of Standard Error in kW *	Timestamp Count *
0.0-0.1	0.0	0	0.0	0
0.1-0.2	0.0	0	0.0	0
0.2-0.3	0.0	0	0.0	0
0.3-0.4	0.0	0	0.0	0
0.4-0.5	4.6	1	0.0	0
0.5-0.6	13.2	4	0.0	0
0.6-0.7	53.7	70	48.0	62
0.7-0.8	-111.2	1478	-48.7	396
0.8-0.9	-551.1	803	-50.8	100
0.9-1.0	0.0	0	0.0	0

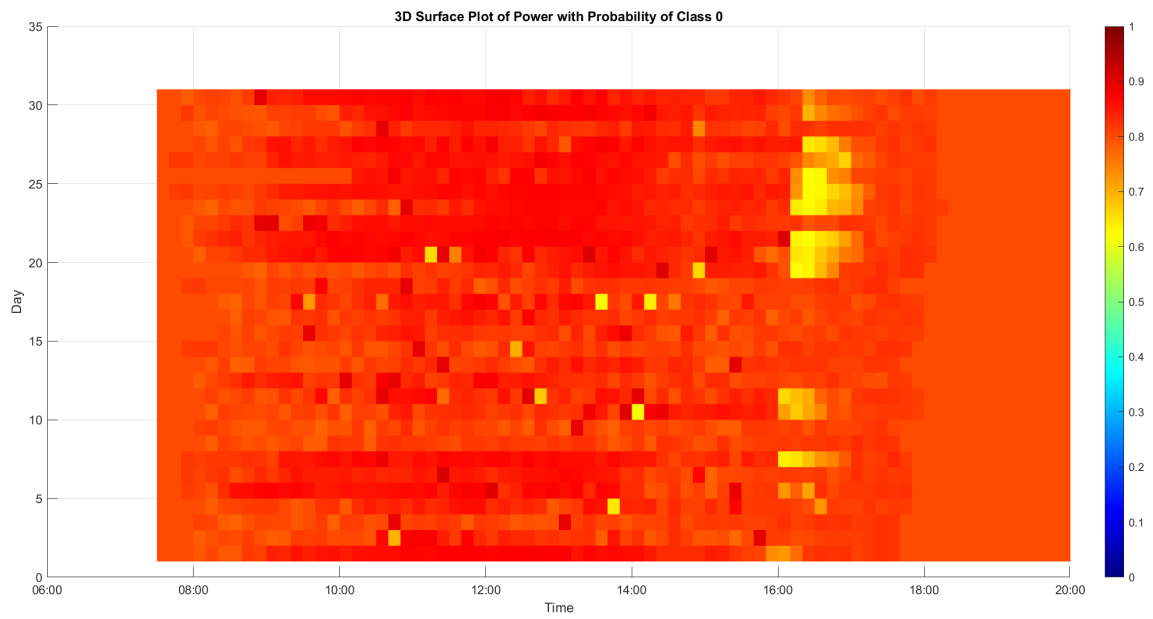


Figure A.32: Model 12

Table A.32: Total power per probability of Class 0 interval for string 15 in January - Model 12

Probability Interval	Sum of Standard Error in kW	Timestamp Count	Sum of Standard Error in kW *	Timestamp Count *
0.0-0.1	0.0	0	0.0	0
0.1-0.2	0.0	0	0.0	0
0.2-0.3	0.0	0	0.0	0
0.3-0.4	0.0	0	0.0	0
0.4-0.5	0.0	0	0.0	0
0.5-0.6	0.0	0	0.0	0
0.6-0.7	55.8	42	34.5	34
0.7-0.8	30.9	778	18.5	117
0.8-0.9	-639.3	1527	-99.0	406
0.9-1.0	-38.3	9	-5.5	1

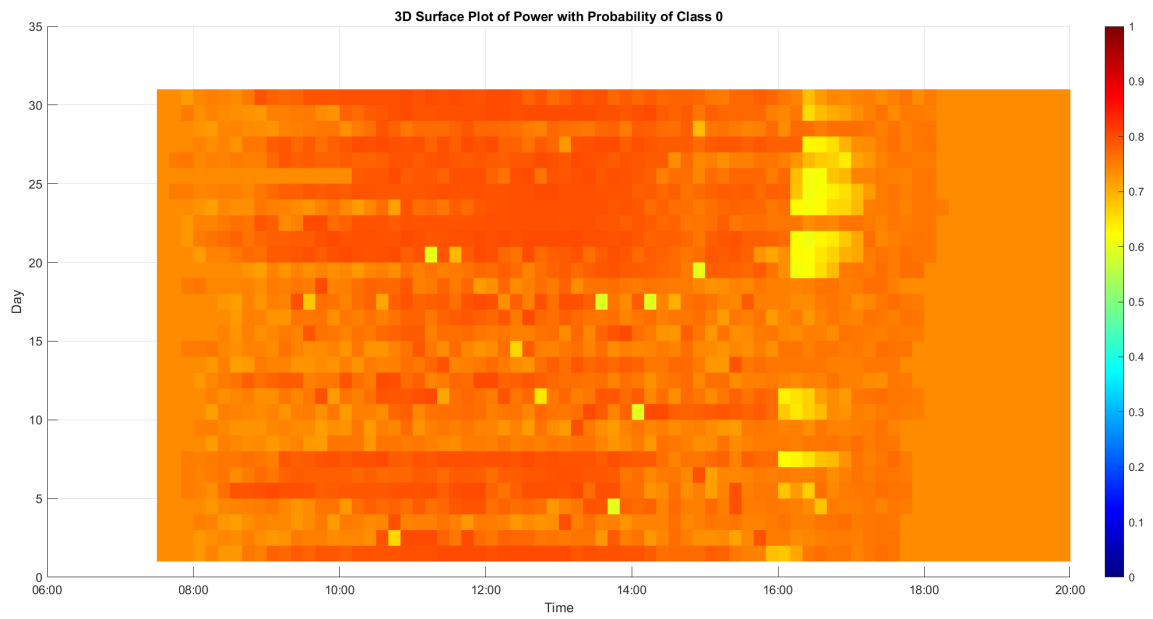


Figure A.33: Model 13

Table A.33: Total power per probability of Class 0 interval for string 15 in January - Model 13

Probability Interval	Sum of Standard Error in kW	Timestamp Count	Sum of Standard Error in kW *	Timestamp Count *
0.0-0.1	0.0	0	0.0	0
0.1-0.2	0.0	0	0.0	0
0.2-0.3	0.0	0	0.0	0
0.3-0.4	0.0	0	0.0	0
0.4-0.5	0.0	0	0.0	0
0.5-0.6	0.0	0	0.0	0
0.6-0.7	55.8	42	34.5	34
0.7-0.8	30.9	778	18.5	117
0.8-0.9	-639.3	1527	-99.0	406
0.9-1.0	-38.3	9	-5.5	1

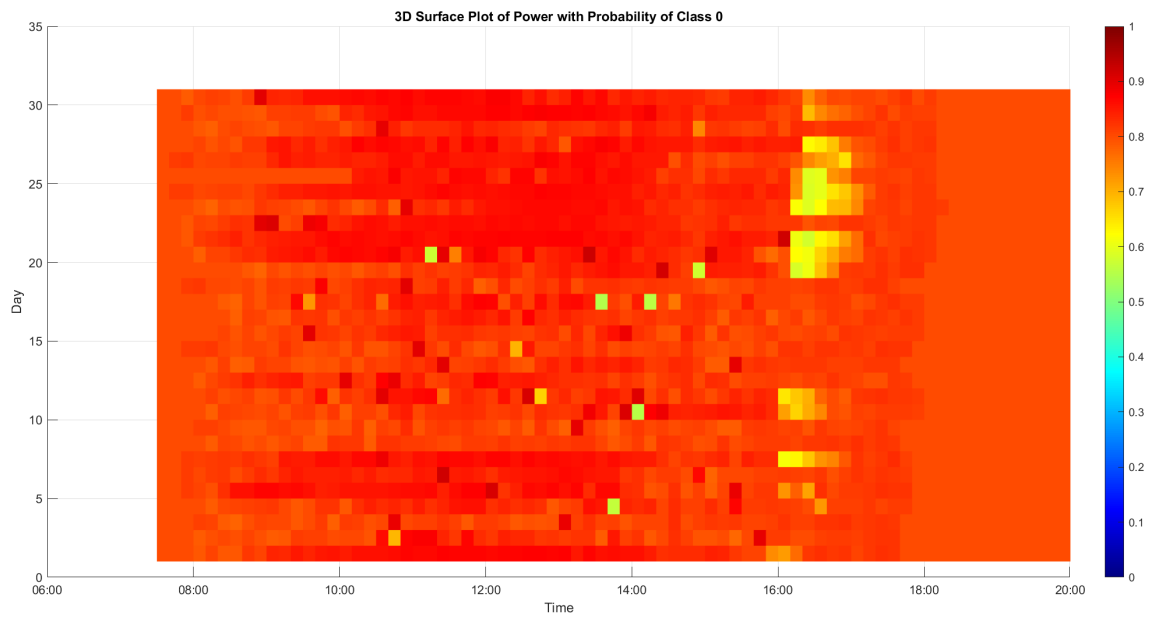


Figure A.34: Model 14

Table A.34: Total power per probability of Class 0 interval for string 15 in January - Model 14

Probability Interval	Sum of Standard Error in kW	Timestamp Count	Sum of Standard Error in kW *	Timestamp Count *
0.0-0.1	0.0	0	0.0	0
0.1-0.2	0.0	0	0.0	0
0.2-0.3	0.0	0	0.0	0
0.3-0.4	0.0	0	0.0	0
0.4-0.5	0.0	0	0.0	0
0.5-0.6	29.1	13	9.4	7
0.6-0.7	27.9	31	25.8	28
0.7-0.8	29.6	781	17.8	119
0.8-0.9	-629.8	1518	-96.5	402
0.9-1.0	-47.7	13	-8.0	2

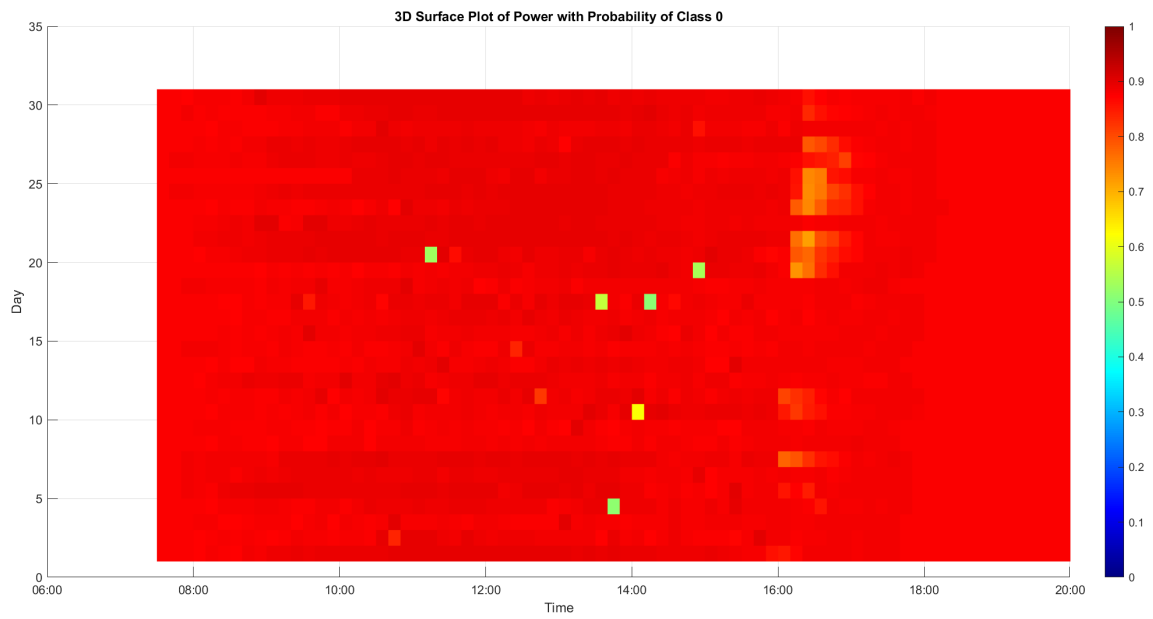


Figure A.35: Model 15

Table A.35: Total power per probability of Class 0 interval for string 15 in January - Model 15

Probability Interval	Sum of Standard Error in kW	Timestamp Count	Sum of Standard Error in kW *	Timestamp Count *
0.0-0.1	0.0	0	0.0	0
0.1-0.2	0.0	0	0.0	0
0.2-0.3	0.0	0	0.0	0
0.3-0.4	0.0	0	0.0	0
0.4-0.5	0.0	0	0.0	0
0.5-0.6	17.8	5	0.0	0
0.6-0.7	1.9	1	0.0	0
0.7-0.8	20.6	17	20.6	17
0.8-0.9	-631.1	2333	-72.5	541
0.9-1.0	0.0	0	0.0	0

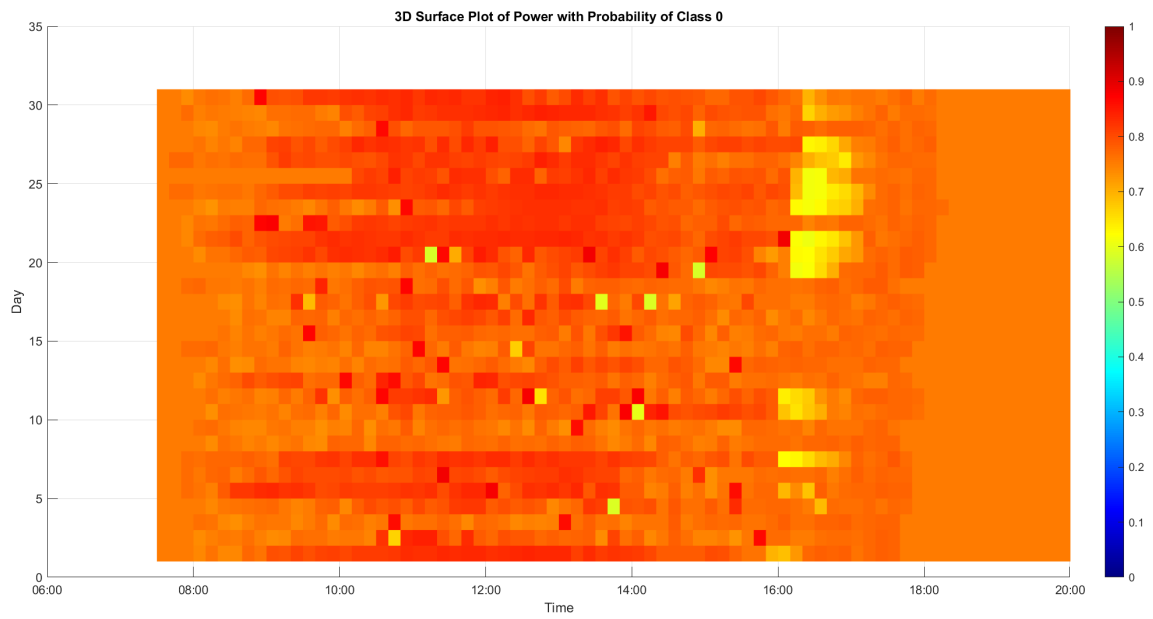


Figure A.36: Model 16

Table A.36: Total power per probability of Class 0 interval for string 15 in January - Model 16

Probability Interval	Sum of Standard Error in kW	Timestamp Count	Sum of Standard Error in kW *	Timestamp Count *
0.0-0.1	0.0	0	0.0	0
0.1-0.2	0.0	0	0.0	0
0.2-0.3	0.0	0	0.0	0
0.3-0.4	0.0	0	0.0	0
0.4-0.5	0.0	0	0.0	0
0.5-0.6	19.7	6	0.0	0
0.6-0.7	48.0	59	45.4	55
0.7-0.8	-203.3	1788	-66.6	471
0.8-0.9	-455.4	503	-30.3	32
0.9-1.0	0.0	0	0.0	0

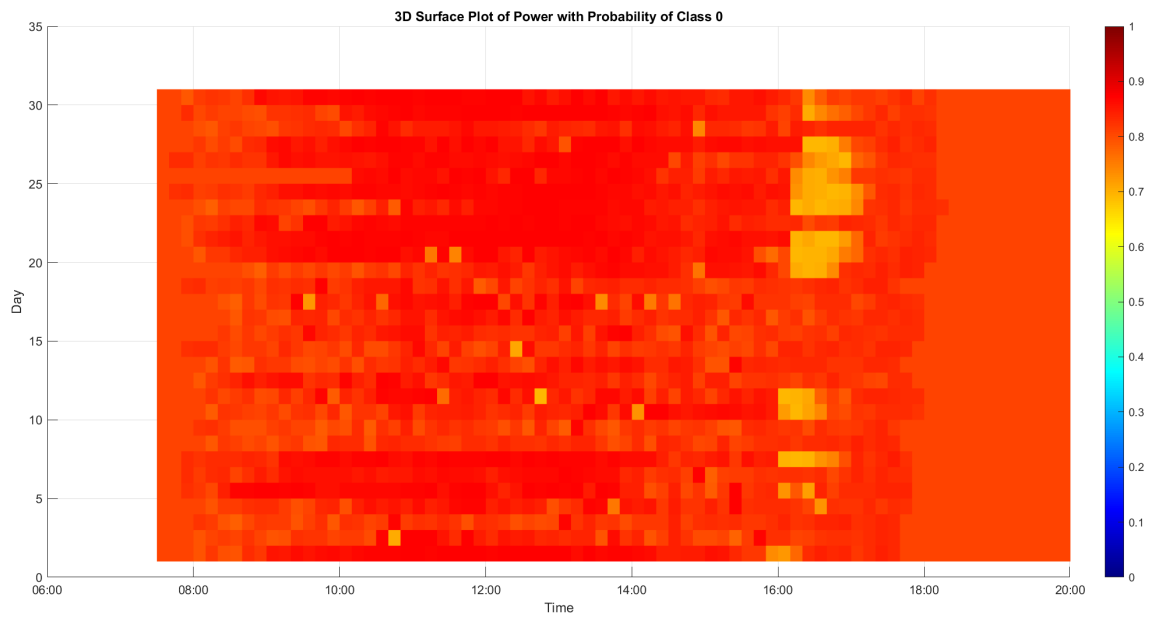


Figure A.37: Model 17

Table A.37: Total power per probability of Class 0 interval for string 15 in January - Model 17

Probability Interval	Sum of Standard Error in kW	Timestamp Count	Sum of Standard Error in kW *	Timestamp Count *
0.0-0.1	0.0	0	0.0	0
0.1-0.2	0.0	0	0.0	0
0.2-0.3	0.0	0	0.0	0
0.3-0.4	0.0	0	0.0	0
0.4-0.5	0.0	0	0.0	0
0.5-0.6	0.0	0	0.0	0
0.6-0.7	23.2	24	22.3	23
0.7-0.8	62.1	177	30.3	73
0.8-0.9	-676.2	2155	-104.1	462
0.9-1.0	0.0	0	0.0	0

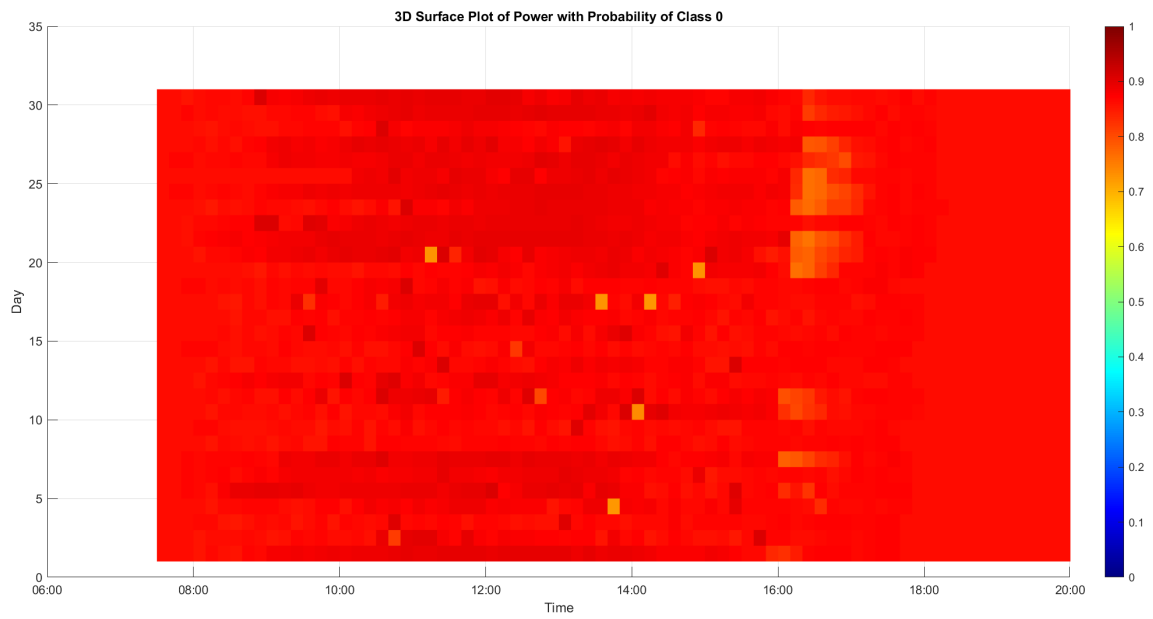


Figure A.38: Model 18

Table A.38: Total power per probability of Class 0 interval for string 15 in January - Model 18

Probability Interval	Sum of Standard Error in kW	Timestamp Count	Sum of Standard Error in kW *	Timestamp Count *
0.0-0.1	0.0	0	0.0	0
0.1-0.2	0.0	0	0.0	0
0.2-0.3	0.0	0	0.0	0
0.3-0.4	0.0	0	0.0	0
0.4-0.5	0.0	0	0.0	0
0.5-0.6	0.0	0	0.0	0
0.6-0.7	0.0	0	0.0	0
0.7-0.8	46.8	30	27.0	24
0.8-0.9	-561.1	2290	-70.0	530
0.9-1.0	-76.5	36	-8.5	4

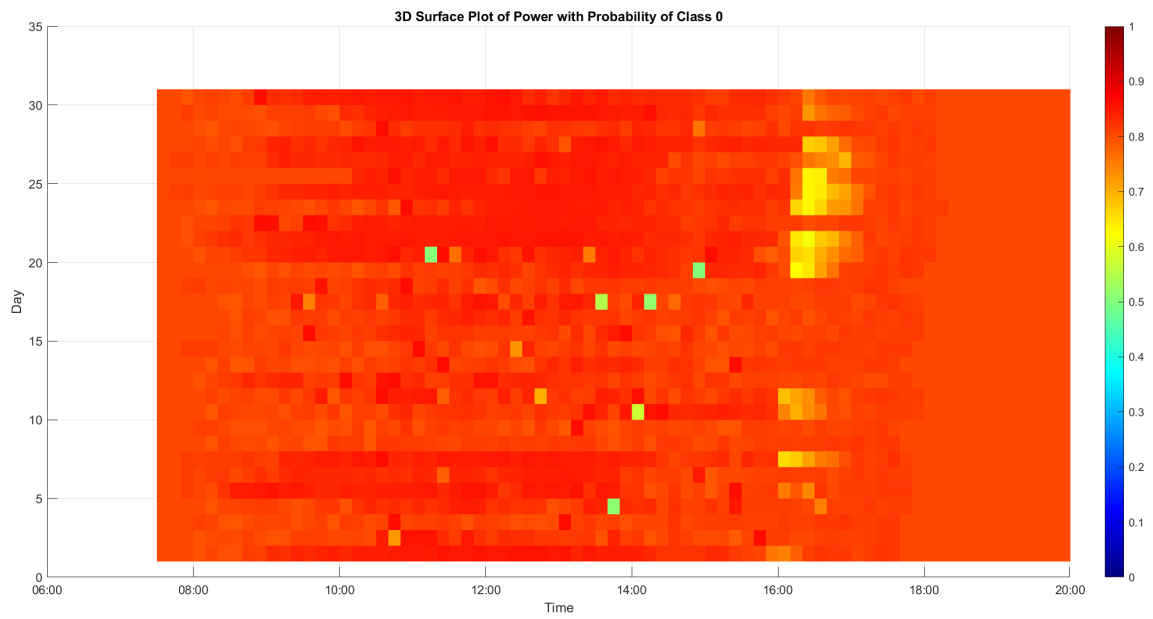


Figure A.39: Model 19

Table A.39: Total power per probability of Class 0 interval for string 15 in January - Model 19

Probability Interval	Sum of Standard Error in kW	Timestamp Count	Sum of Standard Error in kW *	Timestamp Count *
0.0-0.1	0.0	0	0.0	0
0.1-0.2	0.0	0	0.0	0
0.2-0.3	0.0	0	0.0	0
0.3-0.4	0.0	0	0.0	0
0.4-0.5	0.0	0	0.0	0
0.5-0.6	19.7	6	0.0	0
0.6-0.7	27.0	24	27.0	24
0.7-0.8	21.7	222	20.6	87
0.8-0.9	-659.4	2104	-99.1	447
0.9-1.0	0.0	0	0.0	0

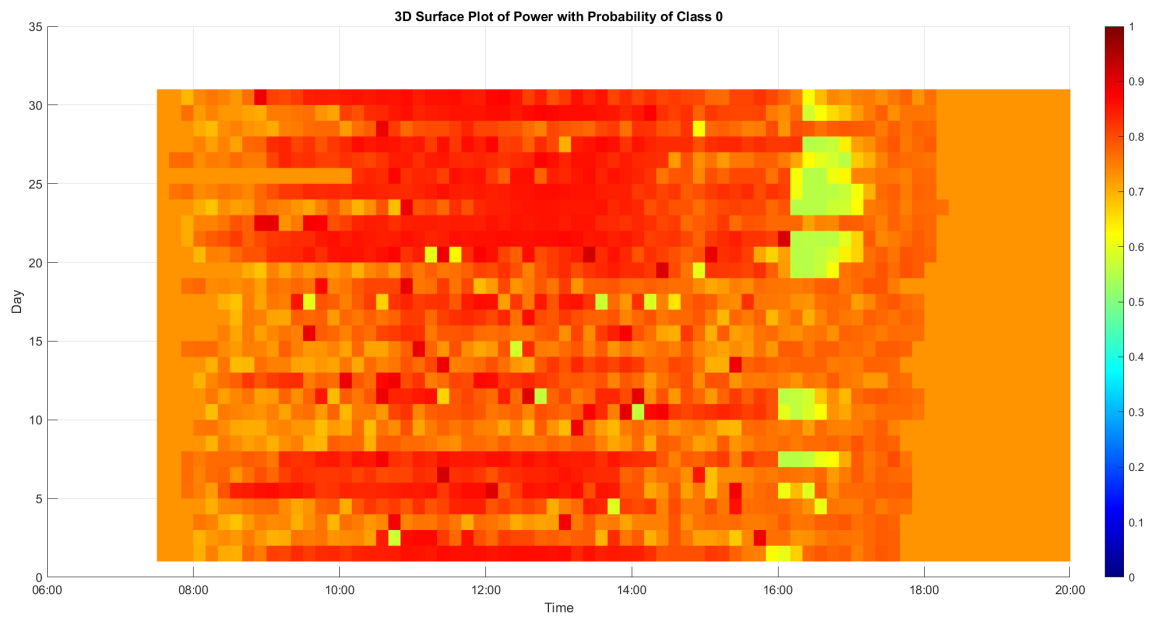


Figure A.40: Model 20

Table A.40: Total power per probability of Class 0 interval for string 15 in January - Model 20

Probability Interval	Sum of Standard Error in kW	Timestamp Count	Sum of Standard Error in kW *	Timestamp Count *
0.0-0.1	0.0	0	0.0	0
0.1-0.2	0.0	0	0.0	0
0.2-0.3	0.0	0	0.0	0
0.3-0.4	0.0	0	0.0	0
0.4-0.5	0.0	0	0.0	0
0.5-0.6	57.7	52	40.4	44
0.6-0.7	21.9	75	11.1	38
0.7-0.8	-144.2	1563	-56.2	406
0.8-0.9	-493.5	659	-41.3	69
0.9-1.0	-32.7	7	-5.5	1