UNIVERSIDAD
DE LA REPUBLICA
URUGUAY

Aprendizaje e inteligencia computacional para la caracterización de consumo eléctrico en hogares

Ignacio Fiori

Mateo Mujica

Mathías Esteban

Programa de Grado en Ingeniería en Computación
Facultad de Ingeniería
Universidad de la República

Montevideo – Uruguay
Noviembre de 2019

UNIVERSIDAD
DE LA REPUBLICA
URUGUAY

Aprendizaje e inteligencia computacional para la caracterización de consumo eléctrico en hogares

Ignacio Fiori

Mateo Mujica

Mathías Esteban

Tesis de Grado presentada al programa de Ingeniería en Computación, Facultad de Ingeniería de la Universidad de la República, como parte de los requisitos necesarios para la obtención del título de Ingeniero en Computación.

Director:
Prof. Sergio Nesmachnow

RESUMEN

Este proyecto estudia el problema de la desagregación energética en hogares, que busca identificar que electrodomésticos se encuentran encendidos en una red eléctrica a partir de los datos de consumo agregado del hogar. Se estudian dos variantes del problema: una variante binaria, donde los electrodomésticos pueden encontrarse únicamente en dos estados, encendido o apagado, y una variante que considera múltiples niveles de consumo.
Se utilizan métodos de aprendizaje automático y redes neuronales para abordar cada una de las variantes. Se implementan cuatro clasificadores: *Naive Bayes*, *K Nearest Neighbours*, *Multi Layer Perceptron*, *Long Short Tert Memory*. Los clasificadores implementados son evaluados utilizando el repositorio de datos UK-DALE. Dicho repositorio contiene información sobre consumo eléctrico de varios electrodomésticos en cinco hogares del Reino Unido en el período entre los años 2012 y 2017. Los resultados experimentales muestran que para la variante binaria, las redes neuronales LSTM son las más apropiadas para abordar el problema de caracterización. Las redes logran tasas de aciertos de cambios de estado de hasta un 75 %, y valores de *f1-score* cercanos a 1. En contraposición, para la variante con múltiples estados de consumo, los resultados son netamente bajos.
Finalmente, se introduce el problema de desagregación de consumo eléctrico a hogares no conocidos. Se prueba el desempeño de los clasificadores frente a conductas de consumo eléctrico no proporcionadas durante las etapas de entrenamiento.

Palabras claves:
desagregación eneregética, aprendizaje automático, inteligencia artificial, redes neuronales, consumo eléctrico.

ABSTRACT

This project focuses on the residential energy disaggregation problem, this problem tries to identify which appliances are on in a electrical network through house aggregate power consumption data. Two variants of the problem are studied: A binary variant, where appliances can be found in only two states, on or off, and another variant which considers multiple levels of consumption. Machine learning methods and neural networks are used to deal with each of the variants. Four classifiers are implemented: *Naive Bayes*, *K Nearest Neighbours*, *Multi Layer Perceptron*, *Long Short Tert Memory*. The implemented classifiers are evaluated using the UK-DALE data repository. This repository holds information about power consumption of several appliances from five United Kingdom's houses in the period between 2012 and 2017 years. Experimental results show that LSTM neural networks are the most appropriate to deal with the characterization problem for the binary variant. These networks achieve success rates of state changes up to 75 % and values of *f1-score* close to 1. On the opposite, for the variant with multiple consumption states, the results are clearly low.
Finally, the problem of energy disaggregation in unknown houses is introduced. The performance of the classifiers is tested against consumption patterns that are not provided during the training stages.

Keywords:
energy disaggregation, machine learning, artificial intelligence, neural networks, power consumption.

Lista de figuras

2.1 Ejemplo de una señal de consumo agregado 4

3.1 Ejemplo de decisión de clasificador KNN según el valor de K. . 16
3.2 Gráfica de la función de activación escalonada. 17
3.3 Gráfica de la función de activación *sigmoid*. 18
3.4 Gráfica de la función de activación tangente hiperbólica. 18
3.5 Gráfica de la función de activación *rectified linear unit*. 19
3.6 Modelo reducido de una red neuronal simple. 22
3.7 Perceptrón 25
3.8 Diagrama perceptrón multicapa. 26

6.1 Últimas cien predicciones de MLP sobre heladera en el escenario 4. 83
6.2 Resultados obtenidos por los clasificadores para el lavavajillas. . 104
6.3 Resultados obtenidos por los clasificadores para la jarra eléctrica. 105
6.4 Resultados obtenidos por los clasificadores para la heladera. . . 106
6.5 Resultados obtenidos por los clasificadores para el microondas. . 107
6.6 Resultados obtenidos por los clasificadores para el lavarropas. . 108
6.7 Resultados para hogares no vistos en entrenamiento obtenidos por los clasificadores para la heladera. 109
6.8 Resultados para hogares no vistos en entrenamiento obtenidos por los clasificadores para la microondas. 110

Lista de tablas

2.1 Ejemplo 1: Tiempo promedio de utilización de cada electrodoméstico en minutos. 7
2.2 Ejemplo 1: Valor consumo *standby* de cada electrodoméstico. . . 7
2.3 Ejemplo 1: Consumo por electrodoméstico día jueves. 7

6.1 Resumen características consideradas en cada escenario. 58
6.2 Ejemplo de matriz de confusión. 62
6.3 Ejemplo de reporte de clasificación. 63
6.4 Descripción de los datos para Naive Bayes en el escenario 1. . . 65
6.5 Resultados para Naive Bayes en el escenario 1. 65
6.6 Métricas de desempeño para Naive Bayes en el escenario 1. . . . 66
6.7 Resultados predichos y reales para Naive Bayes en el escenario 1. 66
6.8 Resultados acertados, reales y porcentaje de aciertos para Naive Bayes en el escenario 1. 66
6.9 Descripción de los datos para Naive Bayes en el escenario 2. . . 67
6.10 Resultados para Naive Bayes en el escenario 2. 67
6.11 Métricas de desempeño para Naive Bayes en el escenario 2. . . . 67
6.12 Resultados predichos y reales para Naive Bayes en el escenario 2. 68
6.13 Resultados acertados, reales y porcentaje aciertos para Naive Bayes en el escenario 2. 68
6.14 Descripción de los datos para Naive Bayes en el escenario 3. . . 69
6.15 Resultados para Naive Bayes en el escenario 3. 69
6.16 Métricas de desempeño para Naive Bayes en el escenario 3. . . . 69
6.17 Resultados predichos y reales para Naive Bayes en el escenario 3. 69
6.18 Resultados acertados, reales y porcentaje aciertos para Naive Bayes en el escenario 3. 70
6.19 Descripción de los datos para Naive Bayes en el escenario 7. . . 70
6.20 Resultados para Naive Bayes en el escenario 7. 71

6.21 Métricas de desempeño para Naive Bayes en el escenario 7. . . . 71
6.22 Resultados predichos y reales para Naive Bayes en el escenario 7. 71
6.23 Resultados acertados, reales y porcentaje aciertos para Naive Bayes en el escenario 7. 71
6.24 Descripción de los datos y mejores parámetros para KNN en el escenario 1 para generalización a hogares vistos durante el entrenamiento. 73
6.25 Tamaños de conjuntos de datos para KNN en el escenario 1 para generalización a hogares vistos durante el entrenamiento. 73
6.26 Resultados para KNN en el escenario 1 para generalización a hogares vistos durante el entrenamiento. 73
6.27 Métricas de desempeño para KNN en el escenario 1 para generalización a hogares vistos durante el entrenamiento. 73
6.28 Resultados predichos y reales para KNN en el escenario 1 para generalización a hogares vistos durante el entrenamiento. 74
6.29 Resultados acertados, reales y porcentaje de aciertos para KNN en el escenario 1 para generalización a hogares vistos durante el entrenamiento. 74
6.30 Descripción de los datos y mejores parámetros para KNN en el escenario 2 para generalización a hogares vistos durante el entrenamiento. 75
6.31 Tamaños de conjuntos de datos para KNN en el escenario 2 para generalización a hogares vistos durante el entrenamiento. 75
6.32 Resultados para KNN en el escenario 2 para generalización a hogares vistos durante el entrenamiento. 75
6.33 Métricas de desempeño para KNN en el escenario 2 para generalización a hogares vistos durante el entrenamiento. 75
6.34 Resultados predichos y reales para KNN en el escenario 2 para generalización a hogares vistos durante el entrenamiento. 76
6.35 Resultados acertados, reales y porcentaje de aciertos para KNN en el escenario 2 para generalización a hogares vistos durante el entrenamiento. 76
6.36 Descripción de los datos y mejores parámetros para KNN en el escenario 3 para generalización a hogares vistos durante el entrenamiento. 77

6.37 Tamaños de conjuntos de datos para KNN en el escenario 3 para generalización a hogares vistos durante el entrenamiento. 77
6.38 Resultados para KNN en el escenario 3 para generalización a hogares vistos durante el entrenamiento. 77
6.39 Métricas de desempeño para KNN en el escenario 3 para generalización a hogares vistos durante el entrenamiento. 77
6.40 Resultados predichos y reales para KNN en el escenario 3 para generalización a hogares vistos durante el entrenamiento. 78
6.41 Resultados acertados, reales y porcentaje de aciertos para KNN en el escenario 3 para generalización a hogares vistos durante el entrenamiento. 78
6.42 Descripción de los datos y mejores parámetros para KNN en el escenario 7. 79
6.43 Tamaños de conjuntos de datos para KNN en el escenario 7. . . 79
6.44 Resultados para KNN en el escenario 7. 79
6.45 Métricas de desempeño para KNN en el escenario 7. 79
6.46 Resultados predichos y reales para KNN en el escenario 7. . . . 80
6.47 Resultados acertados, reales y porcentaje de aciertos para KNN en el escenario 7. 80
6.48 Descripción de los datos para MLP en el escenario 4. 81
6.49 Tamaños de conjuntos de datos para MLP en el escenario 4. . . 81
6.50 Resultados para MLP en el escenario 4. 81
6.51 Métricas de desempeño para MLP en el escenario 4. 82
6.52 Resultados predichos y reales para MLP en el escenario 4. . . . 82
6.53 Resultados predichos, acertados y reales para MLP en el escenario 4. 82
6.54 Resultados de cambios de estado (C-E) reales y acertados para MLP en el escenario 4. 82
6.55 Porcentaje cambios de estado acertados para MLP en el escenario 4. 83
6.56 Descripción de los datos y mejores parámetros para LSTM en el escenario 4 para generalización a hogares vistos durante el entrenamiento. 85
6.57 Tamaños de conjuntos de datos para LSTM en el escenario 4 para generalización a hogares vistos durante el entrenamiento. . 85

6.58 Resultados para LSTM en el escenario 4 para generalización a hogares vistos durante el entrenamiento. 85
6.59 Métricas de desempeño para LSTM en el escenario 4 para generalización a hogares vistos durante el entrenamiento. 86
6.60 Resultados predichos y reales para LSTM en el escenario 4 para generalización a hogares vistos durante el entrenamiento. 86
6.61 Resultados de cambios de estado (C-E) para LSTM en el escenario 4 para generalización a hogares vistos durante el entrenamiento. 86
6.62 Porcentaje de cambios de estado (C-E) acertados para LSTM en el escenario 4 para generalización a hogares vistos durante el entrenamiento. 86
6.63 Descripción de los datos y mejores parámetros para LSTM en el escenario 7 para generalización a hogares vistos durante el entrenamiento. 87
6.64 Tamaños de conjuntos de datos para LSTM en el escenario 7 para generalización a hogares vistos durante el entrenamiento. . 87
6.65 Resultados para LSTM en el escenario 7 para generalización a hogares vistos durante el entrenamiento. 88
6.66 Métricas de desempeño para LSTM en el escenario 7 para generalización a hogares vistos durante el entrenamiento. 88
6.67 Resultados predichos y reales para LSTM en el escenario 7 para generalización a hogares vistos durante el entrenamiento. 88
6.68 Resultados de cambios de estado (C-E) para LSTM en el escenario 7 para generalización a hogares vistos durante el entrenamiento. 88
6.69 Porcentaje de cambios de estado (C-E) acertados para LSTM en el escenario 7 para generalización a hogares vistos durante el entrenamiento. 89
6.70 Descripción de los datos y mejores parámetros para LSTM en el escenario 5 para generalización a hogares vistos durante el entrenamiento. 90
6.71 Tamaños de conjuntos de datos para LSTM en el escenario 5 para generalización a hogares vistos durante el entrenamiento. . 90
6.72 Estados definidos de la heladera para LSTM en el escenario 5 para generalización a hogares vistos durante el entrenamiento. . 91

6.73 Estados definidos del lavarropas para LSTM en el escenario 5 para generalización a hogares vistos durante el entrenamiento. . 91
6.74 Métricas de desempeño de la heladera para LSTM en el escenario 5 para generalización a hogares vistos durante el entrenamiento. 91
6.75 Métricas de desempeño del lavarropas para LSTM en el escenario 5 para generalización a hogares vistos durante el entrenamiento. 91
6.76 Resultados de cambios de estado (C-E) en la heladera para LSTM en el escenario 5 para generalización a hogares vistos durante el entrenamiento. 92
6.77 Resultados de cambios de estado (C-E) en el lavarropas para LSTM en en el escenario 5 para generalización a hogares vistos durante el entrenamiento. 92
6.78 Métricas de desempeño para Naive Bayes en el escenario 3 para generalización a hogares no conocidos. 94
6.79 Resultados sobre cambios de estado para Naive Bayes en el escenario 3 para generalización a hogares no conocidos. 94
6.80 Métricas de desempeño para Naive Bayes en el escenario 7 para generalización a hogares no conocidos. 95
6.81 Resultados sobre cambios de estado para Naive Bayes en el escenario 7 para generalización a hogares no conocidos. 95
6.82 Parámetros óptimos para KNN en el escenario 3 para generalización a hogares no conocidos. 96
6.83 Métricas de desempeño para KNN en el escenario 3 para generalización a hogares no conocidos. 96
6.84 Resultados sobre cambios de estado para KNN en el escenario 3 para generalización a hogares no conocidos. 96
6.85 Parámetros óptimos para KNN en Escenario 7 para generalización a hogares no conocidos. 96
6.86 Métricas de desempeño para KNN en Escenario 7 para generalización a hogares no conocidos. 97
6.87 Resultados sobre cambios de estado para KNN en escenario 7 para generalización a hogares no conocidos. 97
6.88 Métricas de desempeño para LSTM en el escenario 3 para generalización a hogares no conocidos. 98

6.89 Resultados sobre cambios de estado para LSTM en el escenario 3 para generalización a hogares no conocidos. 98
6.90 Métricas de desempeño para LSTM en el escenario 7 para generalización a hogares no conocidos. 98
6.91 Resultados sobre cambios de estado para LSTM en el escenario 7 para generalización a hogares no conocidos. 98
6.92 Ranking de resultados obtenidos por los clasificadores en el lavavajillas. 100
6.93 Ranking de resultados obtenidos por los clasificadores en el microondas. 100
6.94 Ranking de resultados obtenidos por los clasificadores en la heladera. 101
6.95 Ranking de resultados obtenidos por los clasificadores en la jarra eléctrica. 101
6.96 Ranking de resultados obtenidos por los clasificadores en el lavarropas. 102

1.1 Resultados de Naive Bayes en el escenario 1 para la clase apagado. 117
1.2 Métricas de Naive Bayes en el escenario 1 para la clase apagado. 117
1.3 Resultados de Naive Bayes en el escenario 2 para la clase apagado. 118
1.4 Métricas de Naive Bayes en el escenario 2 para la clase apagado. 118
1.5 Resultados de Naive Bayes en el escenario 3 para la clase apagado. 118
1.6 Métricas de Naive Bayes en el escenario 3 para la clase apagado. 118
1.7 Resultados de Naive Bayes en el escenario 7 para la clase apagado. 119
1.8 Métricas de Naive Bayes en el escenario 7 para la clase apagado. 119
1.9 Resultados de KNN en el escenario 1 para la clase apagado. . . 119
1.10 Métricas de KNN en el escenario 1 para la clase apagado. 120
1.11 Resultados de KNN en el escenario 2 para la clase apagado. . . 120
1.12 Métricas de KNN en el escenario 2 para la clase apagado. 120
1.13 Resultados de KNN en el escenario 3 para la clase apagado. . . 120
1.14 Métricas de KNN en el escenario 3 para la clase apagado. 121
1.15 Resultados de KNN en el escenario 7 para la clase apagado. . . 121
1.16 Métricas de KNN en el escenario 7 para la clase apagado. 121
1.17 Resultados de MLP en el escenario 4 para la clase apagado. . . . 122
1.18 Métricas de MLP en el escenario 4 para la clase apagado. 122
1.19 Resultados de redes LSTM en el escenario 4 para la clase apagado. 122

1.20 Métricas de redes LSTM en el escenario 4 para la clase apagado. 123
1.21 Resultados de redes LSTM en el escenario 7 para la clase apagado.123
1.22 Métricas de redes LSTM en el escenario 7 para la clase apagado. 123

Tabla de contenidos

Lista de figuras v

Lista de tablas vi

1 Introducción 1

2 Definición del problema 3

2.1 Introducción . . . 3

2.2 Variante Binaria . . . 5

2.2.1 Formulación matemática . . . 5

2.2.2 Ejemplo . . . 6

2.3 Variante del problema con múltiples niveles de consumo . . . 8

2.3.1 Introducción . . . 8

2.3.2 Definición de estados . . . 8

2.3.3 Complejidades y restricciones . . . 9

3 Marco Teórico 11

3.1 Aprendizaje automático . . . 11

3.1.1 La tarea T . . . 11

3.1.2 La medida de rendimiento P . . . 12

3.1.3 La experiencia E . . . 13

3.2 Naive Bayes . . . 14

3.3 K Nearest Neighbors . . . 15

3.4 Redes neuronales . . . 16

3.4.1 Conceptos generales . . . 17

3.4.2 Perceptrón . . . 24

3.4.3 Perceptrón multicapa . . . 25

3.4.4 Redes recurrentes . . . 26

3.4.5 Convolucionales 30

4 Antecedentes **32**

4.1 Nonintrusive Appliance Load Monitoring - Hart 1992 32

4.2 Dissagregation of Domestic Smart Meter Energy Data 34

4.3 Resumen 36

5 Implementación **37**

5.1 Bibliotecas 37

5.1.1 Numpy 37

5.1.2 Pandas 38

5.1.3 Scikit Learn 39

5.1.4 Tensorflow 39

5.1.5 Keras 40

5.2 Clasificadores implementados para el problema de caracterización energética 40

5.2.1 Naive Bayes 41

5.2.2 K Nearest Neighbors 43

5.2.3 Perceptrón multicapa 45

5.2.4 Long Short Term Memory 45

6 Evaluación experimental **50**

6.1 Repositorios de datos 50

6.1.1 UK-DALE 51

6.2 Definición de los escenarios de evaluación 51

6.2.1 Variante Binaria 53

6.2.2 Variante con mútliples niveles de consumo 57

6.3 Entorno de ejecución 59

6.3.1 Entorno de ejecucón local 59

6.3.2 Entorno de ejecución remoto 59

6.4 Métricas de desempeño 60

6.5 Resultados sobre hogares vistos durante el entrenamiento 64

6.5.1 Naive Bayes 65

6.5.2 K Nearest Neighbors 72

6.5.3 Perceptrón multicapa 80

6.5.4 Redes Neuronales LSTM 84

6.6 Resultados sobre hogares no vistos durante el entrenamiento . . 93

6.6.1 Introducción 93
6.6.2 Naive Bayes 94
6.6.3 K Nearest Neighbors 95
6.6.4 Redes LSTM 97
6.7 Análisis comparativo de los métodos implementados 99

7 Conclusiones y trabajo futuro **111**
7.1 Conclusiones 111
7.2 Trabajo futuro 112

Referencias bibliográficas **114**

Apéndices **116**
Apéndice 1 Resultados de clase apagado para la variante binaria. . 117
1.1 Naive Bayes 117
1.2 K Nearest Neighbours 119
1.3 Multi Layer Perceptron 122
1.4 Long Short Term Memory 122

Capítulo 1

Introducción

El aumento del consumo eléctrico es una tendencia a nivel mundial y cada vez más es el interés por la eficiencia energética. Uruguay no es ajeno a esta realidad, entre los objetivos planteados por UTE para el año 2019 se encuentra la optimización del uso de la energía disponible así como también la incorporación de fuentes energéticas renovables. [1]

El consumo residencial conforma gran parte de la demanda energética de un país, en la búsqueda por mejorar el aprovechamiento de los recursos energéticos es deseable contar con información de consumo desagregada por electrodoméstico. Esta información permitiría la creación de planes de consumo adaptados a diferentes sectores de la población y brindar recomendaciones personalizadas a los usuarios para fomentar el ahorro energético. Siendo aún más ambiciosos, si se contara con la información en tiempo real, se podrían desarrollar mecanismos de encendido/apagado remoto e incluso detectar el funcionamiento erróneo de los electrodomésticos de un hogar. Debido a la dificultad y costo de instalar medidores específicos por cada aplicación en todas las residencias de una localidad, es de interés desarrollar técnicas que sean capaces de brindar estos datos a través de las mediciones agregadas tradicionales.

La caracterización energética es el proceso mediante el cual se busca identificar que elementos provocan una medida agregada de consumo eléctrico. En el caso de hogares, consiste en determinar los electrodomésticos que se encuentran encendidos según el registro de consumo total indicado por su contador. Este tipo de técnicas se basan en el análisis de las firmas particulares de consumo que presentan los electrodomésticos y la conducta rutinaria de las personas que los utilizan.

En el marco de este proyecto se encuentra como objetivo estudiar la desagregación energética y plantear algoritmos de aprendizaje automático que aborden el problema. Como el consumo eléctrico en hogares varía según la región geográfica que se esté estudiando, debido a factores como el clima, el costo de energía eléctrica, la calidad de vida, entre otros, se requieren datos específicos de la zona para testear los algoritmos que se desarrollen.

Como resultado de este proyecto se obtuvieron diferentes contribuciones:

- Análisis del problema y recopilación de las investigaciones previas.
- Diseño, implementación y comparación de diferentes algoritmos de aprendizaje automático.
- Análisis de los conjuntos de datos disponibles y elaboración de una serie de scripts para el procesamiento de los mismos.

Este documento consiste en siete capítulos de los cuales este es el primero y los próximos seis se describen a continuación. En el Capítulo 2 se presenta el problema de manera general y a través de su formulación matemática, se ilustra con un ejemplo y se introduce una variante más compleja del problema. El Capítulo 3 contiene todos los conceptos teóricos que fundamentan este trabajo. En el Capítulo 4 se describen publicaciones previas que abordaron el tema y que sirvieron como base para la producción de esta tesis. El Capítulo 5 presenta las bibliotecas y tecnologías utilizadas para el desarrollo de los algoritmos seleccionados y describe en detalle las implementaciones de cada uno de ellos. En el Capítulo 6 se exponen los resultados obtenidos junto a un análisis de los mismos. Finalmente en el Capítulo 7 se presentan las conclusiones generales del proyecto y posibles tareas futuras.

Capítulo 2

Definición del problema

En este capítulo se presenta el problema de la desagregación energética. En la Sección 2.1 se introduce el tema, se explican los objetivos y los desafíos a superar. En la Sección 2.2.1 se describe los principales lineamientos de la variante binaria del problema finalizando con un ejemplo explicativo. Para finalizar, en la Sección 2.3 se expone la variante que considera múltiples niveles de consumo energético en los electrodomésticos.

2.1. Introducción

La caracterización de consumo energético (en inglés *energy dissaggregation*) refiere al proceso de descomponer una medida agregada de consumo eléctrico en los distintos consumos correspondientes a los electrodomésticos pertenecientes en un hogar. La medida agregada consiste en la demanda acumulada de un conjunto de electrodomésticos en un instante. Si se analiza la carga a lo largo del tiempo se obtiene una señal que varía según las activaciones o apagados del conjunto de electrodomésticos presentes en la casa estudiada.

El problema de desagregación de consumo energético fue introducido por G. W. Hart en su publicación *Nonintrusive Appliance Load Monitoring* en el año 1992 [2]. El objetivo es identificar que electrodomésticos se encuentran encendidos dentro de una red eléctrica a partir de la medida de consumo agregado y de otras variables como pueden ser la hora, el día de la semana e incluso la temperatura. El problema se denomina no intrusivo ya que estas técnicas no implican modificación alguna de los electrodomésticos o la instalación de

nuevos componentes en el tendido eléctrico.

Usualmente los electrodomésticos funcionan a cierto nivel de consumo y se mantienen encendidos durante un determinado período de tiempo, estas características conforman una firma particular. Los algoritmos que atacan este problema buscan identificar estas firmas características en la señal agregada estudiando las variaciones de la misma a lo largo del tiempo. Los métodos de aprendizaje automático resultan especialmente útiles en reconocimiento de patrones por lo que han sido el foco de atención en esta área. La Figura 2.1 presenta una señal de consumo agregado ilustrativa, donde se pueden apreciar variaciones repetitivas y asociadas a electrodomésticos particulares que se desean identificar.

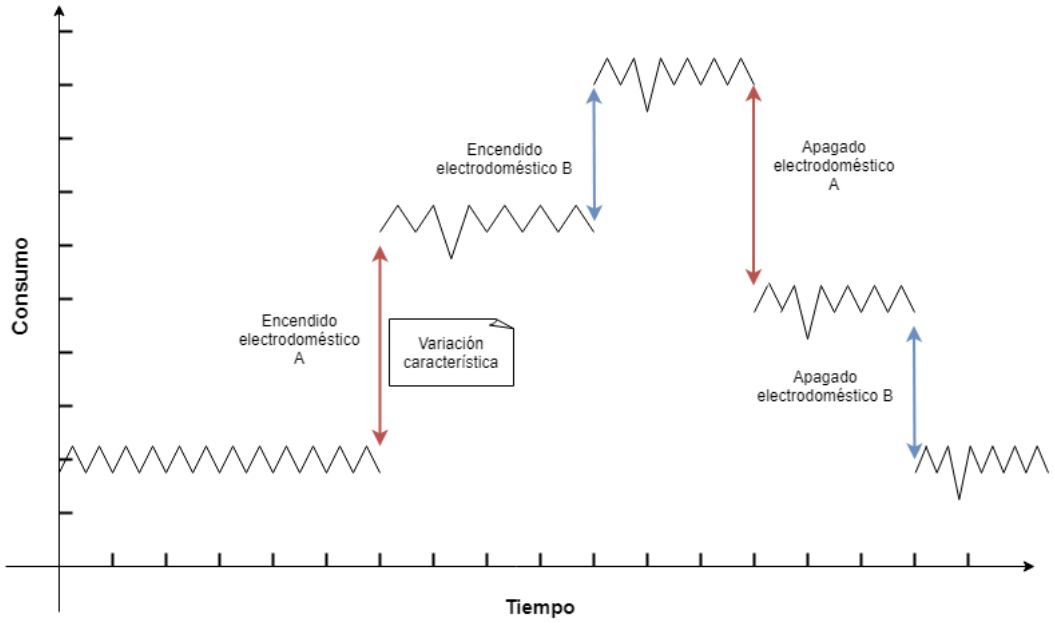


Figura 2.1: Ejemplo de una señal de consumo agregado

En este proyecto el objetivo central es la variante binaria del problema, es decir que los electrodomésticos pueden encontrarse únicamente en dos estados, encendido o apagado. La distinción entre un estado u otro se hace a través de un valor umbral denominado *consumo standby* cuyo valor es específico para cada electrodoméstico. En la Sección 2.3 se introduce una versión alternativa del problema, considerando múltiples estados.

Existen diferentes repositorios que recopilaron registros sobre la demanda agregada y desagregada de electrodomésticos en casas de diferentes partes del

mundo. Estos datos pueden ser utilizados para entrenar algoritmos de aprendizaje, pero existen características inherentes a la naturaleza del problema que hacen dificultoso el análisis de los datos:

- Diferentes electrodomésticos pueden presentar firmas indistinguibles, por esta razón otras variables deben considerarse al momento de decidir si un electrodoméstico se encuentra encendido o no.
- Un electrodoméstico promedio se encuentra apagado la mayor parte del tiempo. Esto implica que los datos recopilados para los estudios se encuentran desbalanceados. Como la información de interés son las fracciones de tiempo durante las cuales el electrodoméstico se enciende, la proporción de datos realmente útil es pequeña. Este desbalanceo de datos hace que sea realmente importante detectar los momentos en los que se produce un cambio de estado.

2.2. Variante Binaria

En esta variante del problema los electrodomésticos pueden clasificarse como encendidos o apagados. En la Subsección 2.2.1 se aborda la formulación matemática, mientras que en la Subsección 2.2.2 se presenta un ejemplo.

2.2.1. Formulación matemática

Sea $E = \{e_1, e_2, ..., e_N\}$ el conjunto de electrodomésticos presentes en un hogar. Dado un instante de tiempo t, el consumo agregado X_t puede especificarse como la Ecuación 2.1.

$$X_t = \sum_{i=1}^{N} a_i(t) y_i + o(t) \tag{2.1}$$

- y_i es el consumo del electrodoméstico e_i.
- $a_i(t)$ es una función booleana que indica si el electrodomésticos e_i se encontraba encendido en el tiempo t.
- $o(t)$ es un factor de ruido que surge de los electrodomésticos no considerados y las señales no medidas.

El objetivo del problema consiste en determinar la función $a_i(t)$ para todos los electrodomésticos involucrados. [3]

2.2.2. Ejemplo

A modo ilustrativo se presenta un ejemplo de la variante binaria del problema. Se plantea una rutina ficticia de utilización de cinco electrodomésticos en un hogar: termofón, lavarropas, televisor, jarra eléctrica y lavavajillas. La rutina diferencia entre días de la semana y fines de semana.

Lunes a viernes:

- 8:00 hs se enciende el termofón
- 20:00 hs se enciende el lavarropas

Lunes y viernes:

- 16:00 hs se enciende el televisor
- 17:00 hs se utiliza la jarra eléctrica

Martes y jueves:

- 19:00 hs se enciende el televisor
- 19:00 hs se utiliza la jarra eléctrica
- 21:00 hs se utiliza el lavavajillas

Sábados y domingos:

- 10:00 hs se enciende el televisor
- 11:00 hs se enciende el termofón
- 16:00 hs se utiliza el lavavajillas

En la tabla 2.1 se ilustran los tiempos promedios de utilización de cada electrodoméstico, mientras que en la tabla 2.2 se observan los valores de *consumo standby* definidos para cada uno. Para valores de consumo debajo del valor de *consumo standby* el electrodoméstico es considerado apagado, mientras que en caso contrario se supone encendido.

Electrodoméstico	Tiempo (minutos)
Termofon	60
Lavarropas	90
Televisor	120
Jarra Eléctrica	10
Lavavajillas	30

Tabla 2.1: Ejemplo 1: Tiempo promedio de utilización de cada electrodoméstico en minutos.

Electrodoméstico	Consumo *Standby* (Wats)
Termofon	50
Lavarropas	50
Televisor	20
Jarra Eléctrica	50
Lavavajillas	20

Tabla 2.2: Ejemplo 1: Valor consumo *standby* de cada electrodoméstico.

Tomando como referencia el día jueves, en la tabla 2.3 se muestra el consumo de la persona entre las 16hs y las 22hs de forma agregada y desglosada por electrodoméstico.

Electrodoméstico	16:00	17:00	18:00	19:00	20:00	21:00	22:00
Termofon	0	0	0	0	0	0	0
Lavarropas	0	0	0	0	50	50	0
Televisor	0	0	0	20	20	0	0
Jarra eléctrica	0	0	0	50	0	0	0
Lavavajillas	0	0	0	0	0	20	0
Total	0	0	0	70	70	70	0

Tabla 2.3: Ejemplo 1: Consumo por electrodoméstico día jueves.

En el ejemplo se observa cómo diferentes conjuntos de electrodomésticos pueden generar los mismos valores de consumo agregado. Será menester de los algoritmos poder diferenciar los subconjuntos en cuestión. Por ejemplo, a las 19hs donde el consumo total es de 70W el algoritmo debe ser capaz de detectar que los electrodomésticos encendidos son el televisor y la jarra eléctrica, mientras que a las 20hs es el lavarropas junto al televisor los que lo están. Al ser día

jueves la persona enciende el televisor a las 19hs y como su uso promedio es de 120 minutos se espera que también lo esté a las 20hs. Por otra parte, como la jarra eléctrica una vez encendida se mantiene así durante aproximadamente 10 minutos puede descartarse como electrodoméstico encendido a la hora siguiente. Por lo tanto, en ciertos casos no basta con estudiar la diferencia de consumo agregado, es necesario apoyarse en otras características para decidir cuáles electrodomésticos se encuentran encendidos y cuáles no. Por ejemplo, algunas de estas características pueden ser el tiempo promedio de activación, la hora, y el día de la semana en cuestión.

2.3. Variante del problema con múltiples niveles de consumo

2.3.1. Introducción

La variante binaria del problema clasifica a los electrodomésticos únicamente como apagado o encendido según su consumo. Sin embargo, existen varios electrodomésticos que trabajan a distintos niveles de potencia. Por ejemplo, cuanto mayor es la diferencia entre la temperatura ambiente y la deseada, mayor va a ser el consumo de un aire acondicionado.
La variante no binaria del problema busca no solo predecir si un electrodoméstico se encuentra encendido en un instante de tiempo o no, sino también saber con que nivel de consumo lo está haciendo. De esta forma se trabaja con firmas más complejas que se asemejan al comportamiento real del electrodoméstico, ya que tienen en cuenta todos sus modos de uso. Las predicciones por lo tanto, son realizadas sobre un conjunto de estados de consumo predefinidos, propios de cada electrodoméstico.

2.3.2. Definición de estados

La primer cuestión a resolver en esta variante del problema es la definición de un proceso que permita hallar los estados de consumo de un electrodoméstico. Se propuso por tanto, un algoritmo que toma como entrada un conjunto de consumos históricos del electrodoméstico y su frecuencia para construir de

manera iterativa los estados de funcionamiento. El algoritmo se presenta a continuación.

El algoritmo toma en cuenta todos los consumos registrados para un electrodoméstico, selecciona aquellos con una cantidad mínima de repeticiones y descarta el resto. Luego ordena los consumos de manera decreciente según la cantidad de apariciones.
Contando con la lista ordenada de consumos, se pasa a la creación de los estados, estos son subdivisiones del intervalo *[0, MAX_CONSUMO]*, donde *MAX_CONSUMO* es el valor de consumo más alto registrado para ese electrodoméstico. Inicialmente se establecen dos estados:

- Estado 1: [0, *standby*]
- Estado 2: [MAX_CONSUMO - *standby*, MAX_CONSUMO]

Se recorre el conjunto de valores de consumo de forma tal que:

- Si el valor de consumo se encuentra comprendido en un estado previamente creado, se descarta.
- Si no se encuentra en un estado ya existente, se debe extender uno ya definido para que lo comprenda o crear uno nuevo. Para tomar una decisión se estudia la intersección del posible nuevo estado con los ya definidos. Si la intersección es vacía el estado se agrega, sino el estado más cercano es extendido para comprender el valor de consumo considerado.

Para finalizar y después de haber recorrido todo el conjunto de consumos, se revisa que no hayan quedado "huecos", pequeños intervalos entre estados que no están comprendidos por ninguno de los estados definidos anteriormente. En estos casos se extienden los estados ya definidos hasta abarcar por completo el "hueco" en cuestión.

2.3.3. Complejidades y restricciones

Las complejidades que presenta la variante de múltiples niveles de consumo son diversas. En primer lugar y al igual que lo que ocurre con la variante binaria, el estado asociado a un electrodoméstico apagado predomina sobre el resto, esto trae aparejado una difícil predicción de los estados que poseen una

cantidad de muestras reducida.
Finalmente, la tarea de definición de estados es compleja. El algoritmo presentado en la subsección previa posee ciertas desventajas a tener en cuenta. La primera y más relevante es que el algoritmo genera un conjunto de estados particular al electrodoméstico del cual se tomaron los datos, volviendo al algortimo específico e incapaz de generalizarse a otros electrodomésticos, aunque sean del mismo tipo.
Otro aspecto negativo es que como aquellos consumos que no logran una cantidad mínima de apariciones son descartados, puede ocurrir que ciertos valores no sean tomados en cuenta a causa de ruidos en la red eléctrica o errores durante la toma de las mediciones. La consecuencia del descarte ocasiona que los consumos identificados como aquellos de mayor frecuencia podrían no serlo realmente.

Capítulo 3

Marco Teórico

En este capítulo se profundiza sobre los aspectos teóricos que forman parte de la línea base de esta tesis. En la Sección 3.1 se introduce el concepto de aprendizaje automático. En las secciones restantes se describe de forma detallada los métodos utilizados y sus fundamentos.

3.1. Aprendizaje automático

Un algoritmo de aprendizaje automático es aquel que tiene la capacidad de aprender determinado comportamiento a partir de los datos que procesa. En el libro *Deep Learning* [4] se proporciona la siguiente definición acerca de que se puede entender por aprender: *"Se dice que un programa de computadora aprende de la experiencia E con respecto a alguna clase de tareas T y la medida de rendimiento P, si su desempeño en tareas en T, medido por P, mejora con la experiencia E"*. En las subsecciones siguientes se describen los puntos más relevantes de la definición anterior.

3.1.1. La tarea T

La tarea es el problema a resolver por el método de aprendizaje automático. Ciertas tareas son tan complejas que serían imposibles de resolver mediante paradigmas tradicionales, en los cuales durante el diseño del programa se especifican las acciones de curso en función de todos los escenarios posibles. Los algoritmos de aprendizaje automático logran aprender el comportamiento deseado en cada situación procesando ejemplos, también denominados muestras. Una muestra es un elemento que contiene la información de entrada ante

la cual debe responder el programa, suele representarse mediante vectores, $x \in R^n$ donde cada componente x_i del vector se denomina característica y aporta información para la resolución del problema. Los tipos de tareas más comunes incluyen:

- **Clasificación:** en este tipo de tareas el método debe encontrar una función que asigne elementos de entrada a un conjunto de categorías.

- **Clasificación con entradas faltantes:** este tipo de tareas es similar al anterior, con la diferencia que algunas de las características de la muestra pueden omitirse, por lo tanto el modelo consiste en hallar múltiples funciones de clasificación, según cual sea la característica faltante.

- **Regresión:** el problema de la regresión consiste en predecir un valor real continuo a partir de los datos de entrada.

- **Transcripción:** en este tipo de tarea se busca identificar texto a través de los datos de entrada.

- **Máquina traductora:** en este tipo de problema los métodos toman la entrada que se encuentra en cierto lenguaje y la traducen a otro.

- **Detección de anomalías:** es un tipo de tarea en el cual se busca detectar valores que no corresponden con patrones previamente aprendidos.

- **Eliminación de ruido:** esta tarea tiene como objetivo recuperar una señal limpia a partir de una contaminada.

3.1.2. La medida de rendimiento P

La medida de rendimiento permite evaluar de manera cuantitativa los resultados obtenidos durante el aprendizaje. Usualmente es una función que recibe como entrada el resultado generado por el algoritmo de aprendizaje y la salida esperada retornando un valor numérico. De esta forma se puede ajustar el modelo buscando minimizar o maximizar los valores arrojados por dicha función según corresponda.

3.1.3. La experiencia E

Los algoritmos de aprendizaje automático pueden clasificarse, según la retroalimentación con la que cuenten durante el proceso de aprendizaje, como supervisados, semisupervisados, no supervisados, o por refuerzo. En lo que respecta a este proyecto se hizo énfasis en la aplicación de los algoritmos supervisados, por lo tanto son abordados en profundidad en las siguientes secciones.

Algoritmos supervisados

En los algoritmos supervisados además de los datos de entrada se cuenta con los datos de salida esperados, por lo tanto el entrenamiento se produce por la corrección de los errores cometidos. Estos algoritmos necesitan de un conjunto de datos etiquetados: ejemplos validados donde a los valores de entrada, los que se conocen como características o (*features*) del elemento a clasificar, se les asocie una salida. El proceso por el cual el modelo evalúa y luego ajusta, según las entradas y salidas de este conjunto de datos, recibe el nombre de entrenamiento. El propio conjunto recibe a su vez la denominación conjunto de entrenamiento.

Un modelo entrenado aún debe ser ajustado, ejecutándolo sobre otro conjunto de datos (no visto durante el entrenamiento) con el objetivo de encontrar la configuración paramétrica que produzca los mejores resultados. Buscar la mejor configuración paramétrica hace referencia al ajuste que se le aplican a los distintos parámetros que conforman al modelo. El conjunto de datos utilizado en esta etapa recibe el nombre de conjunto de validación.

Algunos algoritmos frecuentes de aprendizaje supervisado son: árboles de decisión, Naïve Bayes, regresión por mínimos cuadrados, regresión logística, Support Vector Machines (SVM) y K Nearest Neighbors (KNN).

Las aplicaciones de este tipo de algoritmos son ilimitadas, entre ellas la caracterización de consumo eléctrico. A modo de ejemplo se presentan algunas otras:

- Identificar si un tumor es maligno o no a partir de una imagen del mismo. El conjunto de entrenamiento está formado por imágenes y etiquetas.

Cada imagen de dicho conjunto fue analizada, y la etiqueta indica si el tumor observado es maligno o no.

- Clasificar un email como spam basándose en correos que han sido clasificados previamente de forma manual. El modelo aprende frases y palabras clave que se repiten en los correos no deseados.

3.2. Naive Bayes

Naive Bayes es un algoritmo de clasificación supervisado que utiliza el teorema de Bayes para el cálculo de la probabilidad condicional, con el fin de determinar cual es la clase más probable a la que puede pertenecer un elemento a clasificar.
El término "naive" significa ingenuo y hace referencia a que este método asume la independencia condicional entre cada una de las características presentes en la entrada.
El algoritmo Naive Bayes establece la siguiente ecuación 3.1, donde c es una clase, C es el conjunto de clases predefinidas y e es el elemento a clasificar. c' es la clase con mayor probabilidad teniendo en cuenta las características de e. Por la aplicación del teorema de Bayes la ecuación anterior se puede escribir como 3.2. Si se tiene en cuenta que lo que se busca es maximizar en c y P(e) es un número, entonces no afecta el no tenerlo en cuenta quedando la ecuación reducida a 3.3. Si luego se toma a cada elemento como el conjunto de las características que lo representa, se obtiene la ecuación 3.4. Finalmente, teniendo en cuenta la independencia de las características que conforman a un elemento y reescribiendo la ecuación se llega a 3.5.

$$c' = argmax_{c \in C} \quad P(c|e) \tag{3.1}$$

$$c' = argmax_{c \in C} \quad P(e|c)P(c)/P(e) \tag{3.2}$$

$$c' = argmax_{c \in C} \quad P(e|c)P(c) \tag{3.3}$$

$$c' = argmax_{c \in C} \quad P(x_1, x_2, \ldots, x_n|c)P(c) \tag{3.4}$$

$$cNB = argmax_{c \in C} \quad P(c) \prod P(x|c) x \in X \tag{3.5}$$

Tanto las probabilidades de cada clase P(c) como las $P(x|c)$ se pueden estimar utilizando MAP (Maximum A Posteriori). La estimación MAP es la estimación de una cantidad desconocida que equivale a la moda de la probabilidad posterior de una distribución. La moda por su parte, es el valor que aparece con más frecuencia dentro de un conjunto de valores. Mientras que la probabilidad posterior puede verse como la distribución de probabilidad de una variable aleatoria condicionada por información recabada previamente. Por lo tanto, sea h la cantidad que se desea estimar y sea d la información obtenida previamente, el estimador Map de h se ve en la ecuación 3.6. Si se aplica el teorema de Bayes se obtiene la ecuación 3.7. En este punto podemos prescindir de $P(d)$ debido a que es un valor constante y que se intenta maximizar según h, llegando a la ecuación 3.8. $P(h)$ se conoce como probabilidad a priori.

$$h_{MAP} = argmax_{h \in H} \quad P(h|d) \tag{3.6}$$

$$h_{MAP} = argmax_{h \in H} \quad P(d|h)P(h)/P(d) \tag{3.7}$$

$$h_{MAP} = argmax_{h \in H} \quad P(d|h)P(h) \tag{3.8}$$

3.3. K Nearest Neighbors

K Nearst Neighbours (KNN) es un algoritmo de clasificación que asigna la clase a la que pertenecen la mayor cantidad de los K elementos más cercanos al elemento evaluado.

En problemas de clasificación donde se usa KNN se estila elegir un número de vecinos impar para evitar posibles empates cuando se trabaja en clasificación binaria.

El comportamiento que sigue KNN específicamente es: frente a la llegada de un nuevo elemento a clasificar, calcula la distancia del vector que lo representa hacia todos los vectores conocidos. Luego los ordena de manera creciente según su distancia, para finalmente seleccionar la clase más repetida entre los primeros K elementos de la lista ordenada. Con esta clase clasifica al elemento

en cuestión.
KNN es muy sensible a los valores que se le asignan al número K de vecinos, de modo que con distintos valores se pueden obtener resultados muy dispares. Suponiendo que se quiere clasificar un punto como blanco o negro, tal y como se muestra en la Figura 3.1. Si $K = 1$ la clasificación es que pertenece al conjunto de los puntos blancos. Cuando solo se considera el vecino más cercano puede ocurrir que un punto anómalo guíe al clasificador hacia soluciones erróneas. Para $K = 3$ se asigna al conjunto de los puntos negros.

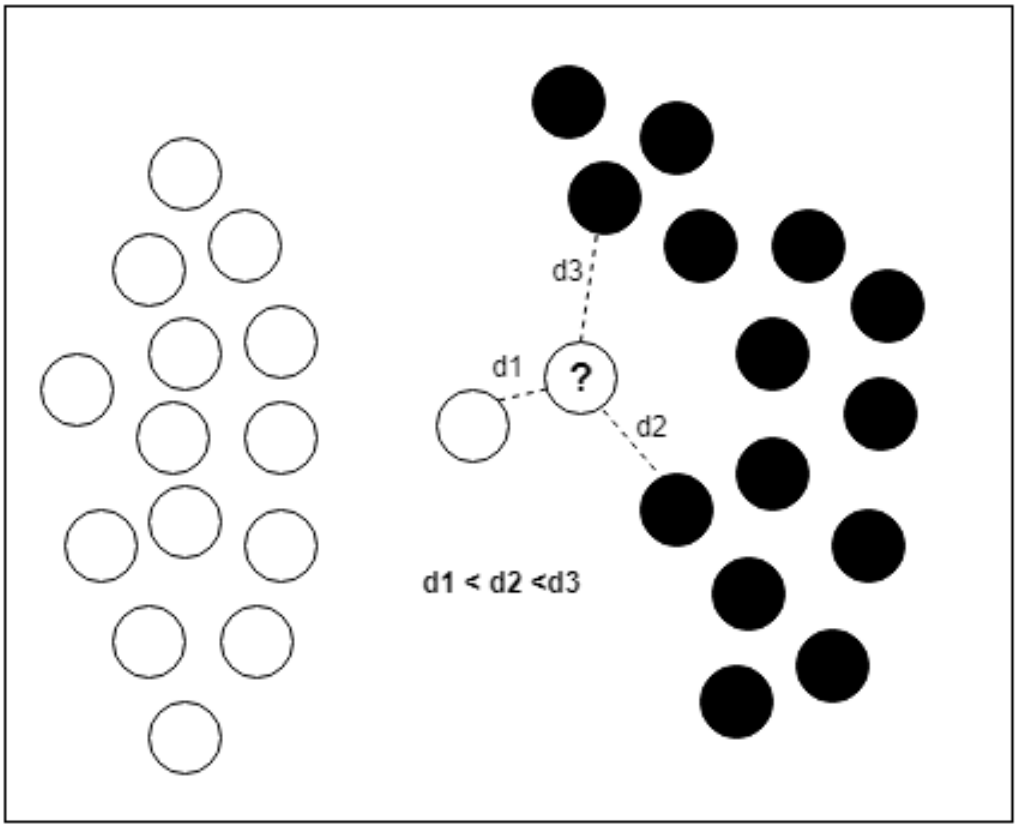


Figura 3.1: Ejemplo de decisión de clasificador KNN según el valor de K.

3.4. Redes neuronales

En esta subsección se presentan los fundamentos teóricos necesarios sobre redes neuronales. Primero se introducen los conceptos generales para luego abordar las diferentes variantes utilizadas.

3.4.1. Conceptos generales

Las redes neuronales artificiales son métodos de aprendizaje automático basados en la estructura biológica de las neuronas. Una neurona recibe una serie de pulsos eléctricos que, si en conjunto son lo suficientemente intensos, provocan la activación. En el cerebro humano la combinación y propagación de activaciones permite tomar decisiones.
En una red neuronal artificial las neuronas son modeladas como funciones matemáticas que operan sobre las entradas que reciben y producen una salida. Las neuronas suman las entradas recibidas y aplican al resultado una función denominada *función de activación* que puede ser discreta o continua.
El desempeño de una red neuronal puede variar significativamente según la función de activación que se elija. Las funciones de activación más utilizadas en la literatura son:

- **Función escalonada**: es la función más simple, tiene una salida binaria, siendo 1 si la entrada a la neurona supera cierto valor predefinido. La definición matemática se puede ver en la Ecuación 3.9 mientras que en la Figura 3.2 se presenta su gráfica.

$$f(x) = \left\{ \begin{array}{ll} 1 & x \geq c \\ 0 & x < c \end{array} \right\} \tag{3.9}$$

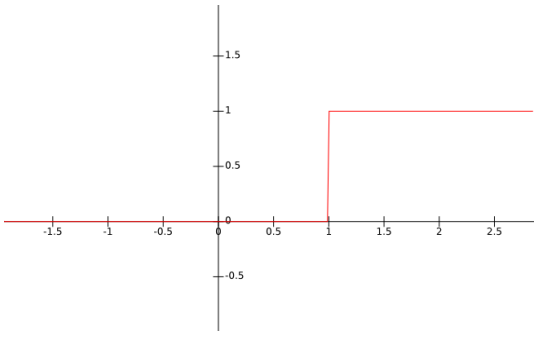


Figura 3.2: Gráfica de la función de activación escalonada.

- **Sigmoid**: es una función continua en el intervalo $(0, 1]$ y su definición se indica en la Ecuación 3.10. La Figura 3.3 muestra una gráfica de la función *sigmoid*.

$$f(x) = \frac{1}{1 + e^{-x}} \tag{3.10}$$

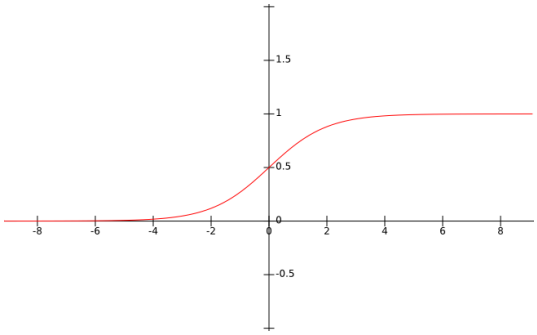


Figura 3.3: Gráfica de la función de activación *sigmoid*.

- **Tangente hiperbólica**: es una función similar a la Sigmoid cuya expresión está dada por la Ecuación 3.11 y su gráfica se puede ver en la Figura 3.4.

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{3.11}$$

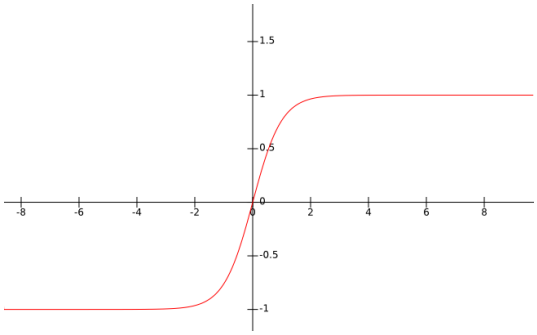


Figura 3.4: Gráfica de la función de activación tangente hiperbólica.

- **ReLU**: esta función retorna valores en el intervalo $[0, +\infty)$ y se define como la Ecuación 3.12. La Figura 3.5 muestra la gráfica de la función *ReLU*.

$$f(x) = \left\{ \begin{array}{ll} x & x > 0 \\ 0 & x \leq 0 \end{array} \right\} \tag{3.12}$$

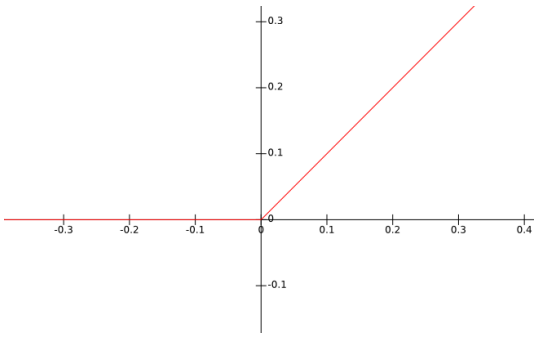


Figura 3.5: Gráfica de la función de activación *rectified linear unit*.

El *bias* es un valor que se suma a la entrada de cada neurona de la red para agregar un grado de libertad. Suele modelarse como una entrada nueva cuyo valor es 1 y su peso sináptico es ajustado durante el entrenamiento. Matemáticamente el *bias* traslada la función de activación sobre el eje de las abscisas.
Las neuronas de una red neuronal se conectan entre sí conformando capas, de forma tal que las salidas de algunas son entradas de otras. Un conjunto especial de neuronas llamado *capa de entrada* recibe los datos a procesar en forma de vector, todas las neuronas de dicho conjunto reciben como entrada todas las componentes del vector. Por otra parte, también se define un conjunto denominado *capa de salida* que determina el resultado de la red.
Las neuronas que no pertenecen a ninguno de los conjuntos anteriores se denominan *ocultas*.
Todas las conexiones entre neuronas son ponderadas, es decir que los valores que se transmiten de una neurona a otra son ajustados según un conjunto de parámetros denominado *pesos sinápticos*.
Mediante un proceso de entrenamiento estos pesos se ajustan de forma tal que la red produzca los resultados deseados. Para llevar a cabo este entrenamiento es necesario contar con un conjunto de vectores de los cuales se conoce el resultado correcto. Una recorrida completa sobre el conjunto de entrenamiento se denomina época (*epoch*). En general son necesarias múltiples recorridas para lograr buenos resultados.
El proceso de aprendizaje funciona de la siguiente forma: de manera iterativa se toma una porción del conjunto de vectores de entrenamiento, la cantidad de elementos seleccionados se denomina *batchsize*. El subconjunto de vectores de entrenamiento es ingresado en la capa de entrada, donde cada neurona aplica su función de activación y propaga el resultado a las neuronas a las cuales esta conectada. La salida propagada por las neuronas es modificada por los

pesos sinápticos antes de ingresar a la próxima neurona. Este procedimiento se aplica sucesivamente en todas las neuronas de la red hasta llegar a la capa de salida.
En este punto la salida de la red se evalúa utilizando una función de costo (*loss function*). Esta función retorna un valor numérico que representa que tan alejado se encuentra el resultado obtenido del deseado. El objetivo de toda red neuronal es minimizar el valor de la función de costo, una mala elección de esta función provoca un mal desempeño de la red. Las funciones de costo más utilizadas son:

- **Error cuadrático:** es una función simple y eficaz que considera el cuadrado de la diferencia entre el valor esperado y el obtenido. Si Y_{real} es el conjunto de n vectores de entrenamiento e Y_{pred} es el conjunto de los n vectores obtenidos por la red. El error cuadrático queda determinado por la ecuación 3.13.

$$E(Y_{real}, Y_{pred}) = \sum_{i=1}^{n} (Yi_{real} - Yi_{pred})^2 \tag{3.13}$$

- **Error cuadrático medio (MSE):** variante de la función anterior donde se toma el promedio del error. Considerando los mismos conjuntos Y_{real} e Y_{pred} de la definición anterior, el *MSE* se define como la ecuación 3.14.

$$MSE(Y_{real}, Y_{pred}) = \frac{1}{n} \sum_{i=1}^{n} (Yi_{real} - Yi_{pred})^2 \tag{3.14}$$

- **Binary Cross Entropy**: un problema de clasificación sobre C clases se puede interpretar con un modelo probabilístico. El objetivo es encontrar una distribución *q(y)* que se asemeje a la distribución verdadera *p(y)*. Para medir la diferencia entre ambas distribuciones se puede utilizar la *entropía*, esta es una medida de la incertidumbre correspondiente a una función de distribución. Su expresión está dada por la Ecuación 3.15.

$$H(p) = -\sum_{c=1}^{C} p(y_c)\ log(p(y_c)) \tag{3.15}$$

Se puede calcular la entropía de una distribución respecto a otra, a lo que se denomina *entropía cruzada* y se define como la Ecuación 3.16

$$H_p(q) = -\sum_{c=1}^{C} p(y_c)\ log(q(y_c)) \tag{3.16}$$

La diferencia entre la entropía y la entropía cruzada se denomina divergencia de *Kullback-Leibler*, y posee la particularidad de ser siempre positiva como se puede ver en la Ecuación 3.17.

$$D_{KL}(p||q) = H_p(q) - H(p) \geq 0 \tag{3.17}$$

Minimizar la divergencia de Kullback-Leibler corresponde a minimizar la entropía cruzada, por lo tanto es factible utilizar la entropía cruzada como función de pérdida. En el análisis del caso binario (C= *2*) llegamos a la expresión indicada por la Ecuación 3.18 donde y es el valor real mientras que $y^{'}$ es el valor predicho.

$$-\left[y\ log(y^{'}) + (1-y)\ log(1-y^{'})\right] \tag{3.18}$$

Los pesos sinápticos deben modificarse de forma tal que el resultado de la función de costo disminuya para vectores futuros. Diferentes algoritmos son utilizados para el ajuste, uno de los más conocidos es el descenso por gradiente, que actualiza los pesos según el vector de derivadas parciales.
Como el objetivo del entrenamiento en una red neuronal es encontrar la configuración de pesos que minimiza el error calculado por la función de costo, el descenso por gradiente es aplicable siendo uno de los más populares. Se calcula el gradiente de la función de costo y se ajustan los pesos hacia la dirección opuesta, una pequeña fracción r denominada *tasa de aprendizaje* como se puede ver en las ecuaciones 3.19 y 3.20. De esta forma, la red se entrena hacia un mínimo (no necesariamente global) de la función de costo.

$$W_{t+1} = W_t + \Delta W_t \tag{3.19}$$

$$\Delta W_t = -r.\nabla E(W_t) \tag{3.20}$$

Existen dos variantes del algoritmo, una variante que se aplica sobre el

conjunto entero de las entradas de entrenamiento y una variante incremental que aplica un procedimiento iterativo sobre los registros. En cada paso de la iteración los pesos se ajustan según las ecuaciones 3.19 y 3.20.

En una red multicapa el error debe ser propagado hacia todas las neuronas a través de las diferentes capas. Cuando se utiliza el descenso por gradiente como algoritmo de optimización, la dificultad reside en el cálculo de las derivadas parciales. Este problema se vuelve computacionalmente complejo a medida que la cantidad de neuronas y capas aumenta.
El algoritmo *Backpropagation* brinda una forma de calcular el vector de derivadas parciales, propone propagar el error desde las capas de salidas hacia las capas de entrada utilizando la regla de la cadena. El algoritmo se vuelve eficiente gracias a la reutilización del cómputo de derivadas parciales.

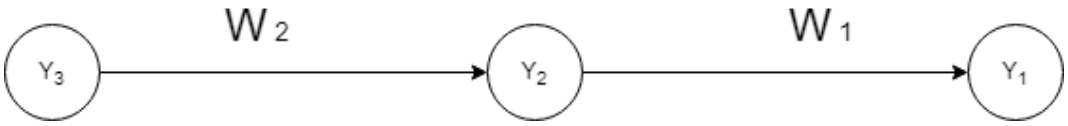


Figura 3.6: Modelo reducido de una red neuronal simple.

A través del ejemplo sencillo presentado en la Figura 3.6 se muestra la fundamentación matemática detrás de este algoritmo, para esto definimos:

$$Z_1 = Y_2.W_1 \tag{3.21}$$

$$Z_2 = Y_3.W_2 \tag{3.22}$$

$$Y_1 = Activacion(Z_1(Y_2, W_1)) \tag{3.23}$$

$$Y_2 = Activacion(Z_2(Y_3, W_2)) \tag{3.24}$$

$$Costo : C(Y_{real}, Y_1) \tag{3.25}$$

El objetivo es hallar las derivadas de la función de costo respecto a todos los pesos sinápticos comenzando en la capa de salida.
Aplicando sucesivamente la regla de la cadena se calcula la derivada de la fun-

ción de costo respecto a los pesos de la última capa llegando a la Ecuación 3.26.

$$\frac{\partial C}{\partial W_1} = \frac{\partial C}{\partial Y_1} \cdot \frac{\partial Y_1}{\partial W_1} = \frac{\partial C}{\partial Y_1} \cdot \frac{\partial Y_1}{\partial Z_1} \cdot \frac{\partial Z_1}{\partial W_1} \tag{3.26}$$

- $\frac{\partial C}{\partial Y_1}$ es la derivada de la función de costo respecto a la salida de la red.
- $\frac{\partial Y_1}{\partial Z_1}$ es la derivada de la función de activación.
- $\frac{\partial Z_1}{\partial W_1}$ es igual a la entrada de la última capa Y_2.

Con la anteúltima capa cuando se quiere hallar la derivada de la función de costos respecto a los pesos sinápticos sucede lo siguiente:

$$\frac{\partial C}{\partial W_2} =$$
$$\frac{\partial C}{\partial Y_1} \cdot \frac{\partial Y_1}{\partial W_2} =$$
$$\frac{\partial C}{\partial Y_1} \cdot \frac{\partial Y_1}{\partial Z_1} \cdot \frac{\partial Z_1}{\partial W_2} =$$
$$\frac{\partial C}{\partial Y_1} \cdot \frac{\partial Y_1}{\partial Z_1} \cdot \frac{\partial Z_1}{\partial Y_2} \cdot \frac{\partial Y_2}{\partial W_2} =$$
$$\frac{\partial C}{\partial Y_1} \cdot \frac{\partial Y_1}{\partial Z_1} \cdot \frac{\partial Z_1}{\partial Y_2} \cdot \frac{\partial Y_2}{\partial Z_2} \cdot \frac{\partial Z_2}{\partial W_2}$$

Se puede ver como aparecen factores ya calculados previamente. Se continúa de manera análoga con las capas siguientes hasta llegar a la entrada de la red.

Existen otros algoritmos de optimización que se pueden utilizar para ajustar los pesos de una red neuronal. *Adam* [5] es un algoritmo de optimización diseñado con el propósito de ser utilizado en el entrenamiento de redes neuronales profundas. Su aparición es reciente, data del año 2014 y se destaca porque a diferencia del descenso por gradiente tradicional, tiene una tasa de aprendizaje adaptativa. Esto quiere decir, que se mantiene una tasa de aprendizaje individual para cada peso de la red, la cual se adapta conforme sucede

el entrenamiento.
Adam toma las ventajas de otros dos algoritmos de optimización, extensiones del descenso por gradiente, que se conocen como *AdaGard* (*Adaptive Gradient Algorithm*) y *RSMprop* (*Root Mean Square Propagation*). Del primero recoge que mantiene la tasa de aprendizaje por parámetro, lo cual mejora el rendimiento en problemas de gradientes dispersos. Sin embargo, en determinadas circunstancias puede suceder que las tasas de aprendizaje disminuyan hasta ser casi nulas, lo que impide que el modelo aprenda. *RSMprop* introduce algunas características que corrigen el problema antes mencionado y son estas la que adopta el optimizador *Adam*. Se convirtió en uno de los optimizadores por defecto para el entrenamiento de redes neuronales.
En las subsecciones siguientes se presentan los modelos de redes neuronales utilizados en la evaluación y que se describen en la Sección 6.

3.4.2. Perceptrón

A modo de introducción se presenta el modelo computacional básico de una red neuronal, el Perceptrón. Consiste en una única neurona, donde a cada conexión de entrada se le asigna un peso adaptativo como se muestra en la figura 3.7. Si la suma de las entradas ponderadas por los pesos definidos es mayor a cierto umbral se produce una activación.

Si se interpreta la entrada como el vector $X = (x_0, x_1, x_2, ..., x_n)$, los pesos como el vector $W = (w_0, w_1, w_2, ..., w_n)$, y el umbral como el valor escalar b, la salida del perceptron queda determinada por la Ecuación 3.27.

$$perceptron(n) = \begin{cases} 1 & \sum_{i=0}^{n} x_i.w_i > b \\ 0 & \sum_{i=0}^{n} x_i.w_i \leq b \end{cases} \tag{3.27}$$

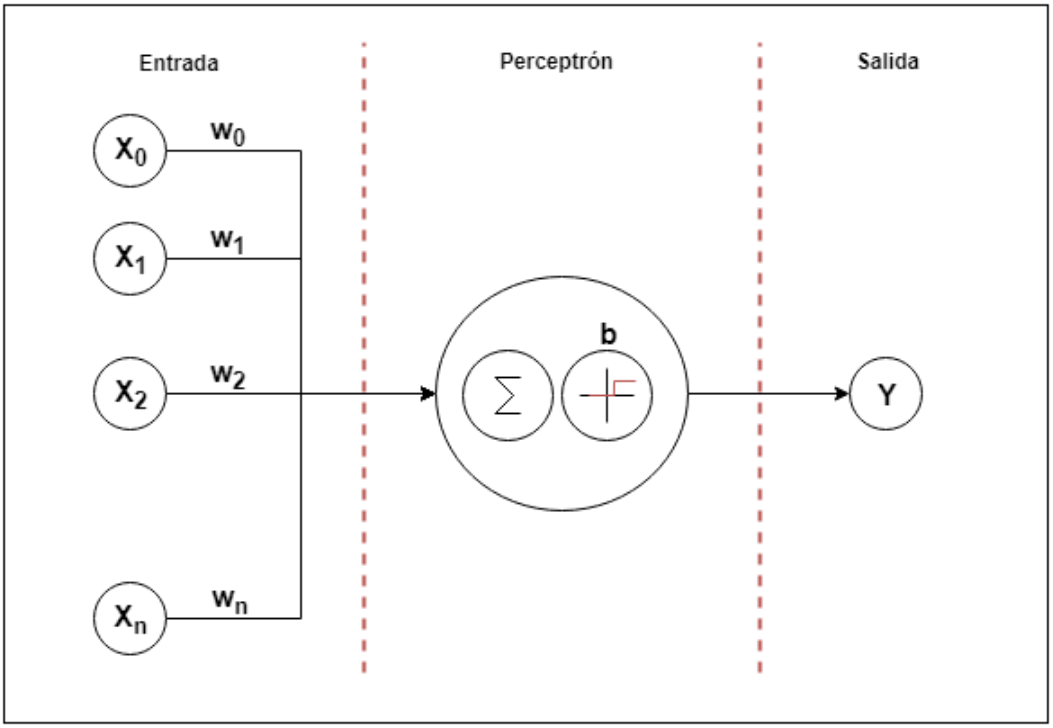


Figura 3.7: Perceptrón

El aprendizaje consiste en hallar el vector W y el valor b que generen las activaciones correctas para el problema que se intenta modelar.
Se puede observar que un perceptrón es un clasificador lineal, por lo que será capaz de resolver aquellos problemas donde el conjunto de datos sea linealmente separable. [6]

Ante este dilema, surge la necesidad de desarrollar estructuras más complejas donde la cantidad de neuronas aumenta, y las interconexiones entre ellas conforman capas entre las entradas y las salidas de la red.
Por ejemplo las redes de perceptrones multicapa también conocidas como MLP (*Multi Layer Perceptrón*) son una agrupación de perceptrones donde solo son permitidas las conexiones hacia adelante. Por otra parte, aquellas redes donde las conexiones hacia atrás son posibles se denominan redes recurrentes. Existen otros tipos de redes neuronales que son explicados en las secciones siguientes.

3.4.3. Perceptrón multicapa

El perceptrón multicapa es una red de perceptrones donde solo se permiten conexiones hacia adelante. Esta condición lleva a conformar capas bien distinguidas dentro de la red tal como se muestra en la Figura 3.8 . Formalmente en

el problema de clasificación, dado $y = g(x)$ que asigna x a una categoría y, el MLP define una función $y = f(x; O)$, donde el conjunto de parámetros O es el que se debe aprender de forma tal que f aproxime de la mejor manera posible a g. [4]

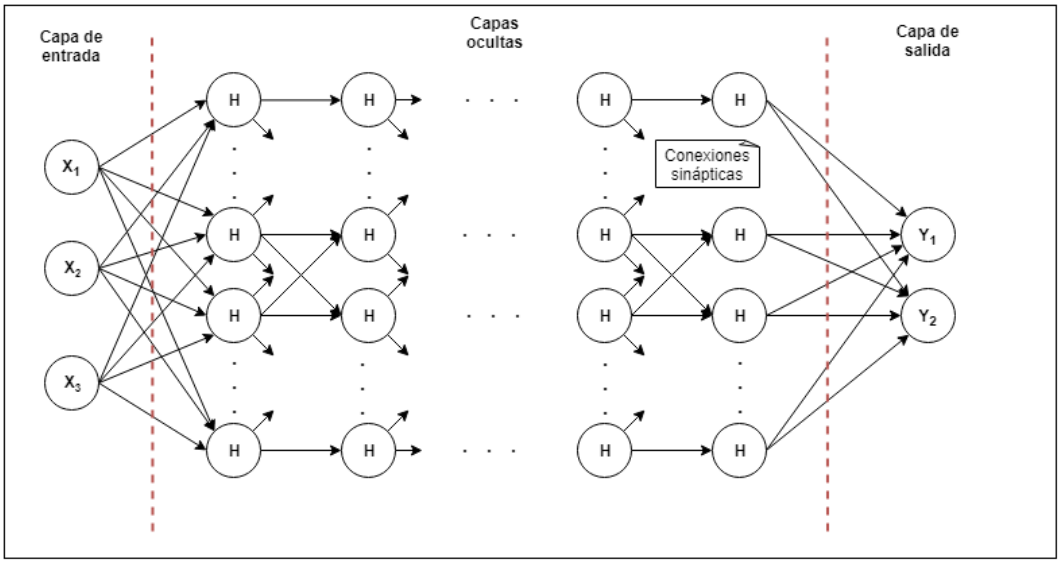


Figura 3.8: Diagrama perceptrón multicapa.

La primer capa de la red se denomina de entrada mientras que la última es la de salida, las capas intermedias se denominan ocultas. Cada capa oculta es una función, y la composición de todas aplicada sobre los datos de entrada produce la salida de la red.
Tanto la profundidad total de la red neuronal como la dimensión de cada capa son criterios de diseño, en particular cuando la cantidad de capas ocultas es mayor a uno se utiliza el término *Deep Learning*.

3.4.4. Redes recurrentes

Introducción

Las redes neuronales recurrentes (RNN por su sigla en inglés) son un tipo especial de redes diseñadas para abordar problemas de secuencia, es decir problemas donde las entradas tienen cierta dependencia temporal.
La idea principal es almacenar los valores generados por cada una de las neuronas y así, contar con esa información para determinar la salida en próximas iteraciones, de las propias neuronas o de otras presentes en diferentes capas de

la red.
La arquitectura que siguen este tipo de redes, se puede pensar como la de un perceptrón multicapa al que se le agregan ciclos entre las neuronas de las distintas capas.
Las redes recurrentes mantienen memoria, que suele implementarse teniendo neuronas de estados ocultos (hidden state) donde para determinado tiempo t, su salida es calculada mediante la información almacenada en un tiempo $t-1$ conjuntamente con la entrada del vector actual. El proceso anterior se describe matemáticamente por la Ecuación 3.28

$$h_t = \phi(W * x_t + U * h_{t-1}) \tag{3.28}$$

Cada uno de los factores de la ecuación 3.28 representa lo siguiente:

- ϕ es una función de activación no lineal como puede ser el caso de *Relu* o *Sigmoid*.
- h_t es el valor del estado oculto en el paso t.
- x_t es el valor del vector de entrada.
- W es la matriz de pesos.
- U es otra matriz que se conoce como matriz de transición y es similar a una cadena de Markov.
- h_{t-1} es el valor del estado oculto en el paso t-1.

Son los estados ocultos los encargados de capturar la potencial relación entre los distintos vectores de una serie. Esta relación se irá modificando en cada paso del entrenamiento, a medida que las cualidades más antiguas se van olvidando.
Esta memoria es la que luego será tomada como entrada para cálculos en capas posteriores. Lo que trae como consecuencia que la red aprenda dependencias a largo plazo (*long-term dependences*) en una secuencia, y esto es muy beneficioso a la hora de realizar una predicción donde la salida dependa de contextos anteriores.

Las redes neuronales recurrentes son muy utilizadas en el ámbito del procesamiento de lenguaje natural, clasificación de sentimientos, reconocimiento de series o secuencias entre otros.

Entrenamiento de una RNN

Uno de los principales inconvenientes que tuvieron que enfrentar los precursores de las redes recurrentes fue identificar de qué forma modificar el método clásico de entrenamiento para adaptarlo a estas topologías. El algoritmo de *Backpropagation* que se basa en propagar hacia atrás el error de predicción para luego modificar los pesos sinápticos, falla en el momento en que se encuentra con los bucles de las redes recurrentes. Surgió entonces una modificación a esta técnica conocida con el nombre de *Backpropagation Through Time* ($BPTT$) con el fin de poder solucionar los problemas del método original. Lo que plantea $BPTT$ es hacer modificaciones en la estructura cíclica introduciendo nuevas neuronas donde se producen conexiones hacia atrás, estas neuronas comparten los mismos pesos que las originales. De este modo, se logra transformar el grafo cíclico en una red *feedforward*, donde luego se aplica *Backpropagation* de la forma tradicional.

RNN Profundas

Al igual como sucede con las redes tradicionales, aumentar la profundidad puede mejorar los resultados obtenidos en ciertos problemas. Una manera de hacerlo consiste en incrementar el número de estados ocultos sucesivos, alimentándose uno con la salida del otro. Otra alternativa es agregar capas ocultas no lineales entre la capa de entrada y la capa de salida.
La principal ventaja que otorgan las redes recurrentes profundas es una capacidad de aprendizaje muy alta pero requiere como contrapartida una gran cantidad de datos para poder entrenarla.

RNN Bidireccionales

Esta variante de las redes recurrentes está basada en que la salida en un tiempo t no solamente depende de los elementos previos de una secuencia, sino también de los elementos futuros. Lo mismo que ocurre por ejemplo en NLP (*Natural Language Processing*) donde una palabra depende tanto de lo que está detrás de ella como lo que está por delante.
El principal desafío de las redes recurrentes bidireccionales es saber hasta qué punto observar en el futuro, ya que si se espera a poder ver todas las entradas la operación se vuelve realmente costosa.
Finalmente, las redes neuronales bidireccionales son dos RNN apiladas una

encima de otra. La salida es calculada según lo establecido por los estados ocultos de ambas RNN.

Long Short Term Memory (LSTM)

Dentro de las redes neuronales recurrentes, las que han adquirido mayor relevancia son las conocidas con el nombre de *Long Short Term Memory* (*LSTM*).
Las redes recurrentes *LSTM* surgen con el objetivo de poder solucionar los problemas que evidencian las redes recurrentes convencionales en lo que respecta a su entrenamiento. El principal inconveniente que presentan está relacionado con la retropropagación de los gradientes. Los gradientes tienden a aumentar de manera significativa o a desaparecer con el paso del tiempo debido a que el gradiente no depende solo del error actual sino de los acarreados desde el pasado (*Vanishing Gradient Problem*).
Las neuronas de una red LSTM son "células" (*cells*) que se encargan de guardar información al igual que una memoria. Toman como entradas lo calculado en el estado anterior y la entrada del vector y mediante un método de caja negra deciden qué guardar y qué borrar de la memoria. Luego de esto, las neuronas calculan la salida con lo guardado actualmente en la memoria, la entrada del vector y lo calculado en el paso anterior.
La información que se agrega o remueve de la célula está cuidadosamente regulada por ciertas estructuras que reciben el nombre de puertas (*gates*). Dichas puertas permiten saber qué información de la que estaba anteriormente almacenada se va a utilizar, que de la actual se va a guardar y que de lo calculado en este período va a ser retornado como el resultado. Existen tres tipos de puertas las que se clasifican de la siguiente manera:

- **Entrada**: encargada de controlar qué información nueva puede entrar en la memoria y en qué momento puede hacerlo.
- **Olvido**: encargada de decidir qué información debe olvidar la memoria y en qué momento.
- **Salida**: encargada de controlar cuando se utiliza el resultado de la memoria almacenado, en la celda.

3.4.5. Convolucionales

Las redes neuronales convolucionales (CNN de su sigla en inglés) son un tipo de red neuronal particular creada con el objetivo de trabajar sobre imágenes, especialmente imágenes representadas como matrices de enteros. Son muy utilizadas porque, a diferencia de otros algoritmos de *Machine Learning* donde se requiere un trabajo pesado para descubrir y definir las características, son capaces de aprender dichas características de forma automática, entrenamiento mediante. A su vez, tienen la capacidad de ponderar ciertos constitutivos de los objetos que se encuentran en una imagen y poder diferenciarlos entre ellos.

El elemento central y más importante de la arquitectura de una red convolucional, es la capa convolucional (*Convolutional Layer*) que le da el nombre a la misma. Esta capa es la encargada de aplicar una operación llamada convolución. Hablando en términos de redes neuronales, una convolución se define como una multiplicación de matrices, más precisamente el "producto punto" o "producto escalar" entre ellas. Las matrices involucradas son la matriz de entrada (usualmente una parte de la información de una imagen representada mediante una matriz), y un filtro o "detector de características".

Un filtro no es más que un conjunto de pesos dispuestos en forma de matriz. El tamaño de los filtros siempre se establece menor que el de la matriz de entrada y esto permite que la multiplique en distintas partes o sectores. La aplicación de un filtro (convolución) sobre un sector de la entrada da como resultado un elemento de una nueva matriz que recibe el nombre de "Mapa de Activación" o "Mapa de Características". El resto de los valores que integran esta matriz se calculan repitiendo esta operación sobre el resto de los sectores. El objetivo que busca la aplicación de filtros es poder detectar características, y en la práctica las redes neuronales aprenden los valores de los filtros durante el entrenamiento por cuenta propia.

Una práctica muy utilizada luego de conformado el mapa de activación es agregarle no linealidad a sus integrantes. Esto se alcanza mediante la aplicación de la función *ReLU* (*Rectified Linear Unit*) a cada uno de los componentes. Se aplica con el fin de representar mejor la realidad, ya que la información con la que suele trabajar una CNN es no lineal.

Otro componente importante de la arquitectura de una CNN es la capa de agrupación (*Pooling Layer*) que tiene la función de reducir la dimensión del

mapa de activación, sin perder las características más importantes. El principal objetivo de esta capa es el de reducir el poder computacional necesario para tratarlo. Dentro de los tipos de agrupación los más utilizados son *MaxPooling*, *SumPooling* y *AvgPooling* donde se queda con el valor máximo, se suman todos los valores, o se promedian los valores del sector o ventana a reducir respectivamente.

Para finalizar el proceso, se pasa la salida obtenida hasta el momento a una nueva capa *Flatten Layer* con el objetivo de transformarla en un vector de entrada para la capa Totalmente Conectada (*Fully Connected Layer*). Dicha capa, que implementa usualmente un perceptrón multicapa con una función de activación *SoftMax* o *Sigmoid* en la capa de salida, tendrá asignada la tarea de clasificar el vector de entrada.

Capítulo 4

Antecedentes

Diversos investigadores han abordado el problema de la desagregación energética, contribuyendo cada uno con interesantes aportes. Para el desarrollo de esta tesis el enfoque estuvo fuertemente concentrado en dos estudios. El primero consiste en el trabajo pionero de Hart, quien en el año 1992 publicó su artículo "Nonintrusive Appliance Load Monitoring" [2]. En su artículo, Hart introdujo el problema de la desagregación energética junto a algunas definiciones que fueron base para todos los trabajos posteriores. El segundo antecedente a destacar es la investigación llevada a cabo por Kelly en el año 2016, como parte de su tesis doctoral en la Universidad de Londres y bajo la supervisión de Knottenbelt [7]. En este capítulo se presentan los elementos principales desarrollados por ambos antecedentes.

4.1. Nonintrusive Appliance Load Monitoring - Hart 1992

Hart presentó el concepto NALM describiéndolo como una técnica para determinar el consumo individual de los electrodomésticos que se encienden y apagan en una carga eléctrica.
Hart describió dos variantes de NALM que se diferencian en el grado de intrusión sobre los electrodomésticos de la red. La primera es denominada *Manual Setup* (MS-NALM) y requiere de un proceso de recopilación y clasificación previo de las firmas de consumo. Para lograr este proceso de recopilación y clasificación, se encienden y apagan los electrodomésticos de forma manual observando las variaciones que se producen.

La segunda variante de NALM que introdujo Hart es llamada *Automatic Setup* (AS-NALM), que no requiere una etapa intrusiva de observación, pero si es necesario contar con información relevante a los electrodomésticos involucrados. En el artículo también se desarrollan tres modelos sobre el comportamiento de los electrodomésticos: ON/OFF, *Finite State Machine* y *Continuously Variable*.

El modelo ON/OFF fue el utilizado en esta tesis. El modelo es aplicable a la mayoría de los electrodomésticos de un hogar, aunque no se adapta de la mejor manera a aquellos que se caracterizan por tener múltiples niveles de consumo.

El modelo de estados fue presentado en el Capítulo 2. Este modelo considera múltiples niveles de consumo por electrodoméstico, volviéndose fundamental identificar las transiciones entre los diferentes estados. Es un modelo más realista que el ON/OFF pero aumenta la complejidad de los algoritmos que intentan resolverlo.

Finalmente, el tercer modelo presentado por Hart es el continuo. Este modelo incorpora aquellos electrodomésticos que no presentan rangos fijos de consumo. Hart fundamentó además, que este comportamiento no es el seguido por los electrodomésticos residenciales, por lo cual este modelo es de más interés en el mundo comercial e industrial.

Otro de los aportes que se pueden encontrar en el artículo de Hart es la taxonomía de firmas. En una primera clasificación la taxonomía distingue aquellas firmas no intrusivas de las que si lo son. Las firmas no intrusivas son las que se pueden obtener mediante la observación de la carga agregada y se clasifican en firmas de estado estacionario (*steady state*) o transitorias (*transient*). Una firma de estado estacionario consiste en características propias del estado en el que opera un dispositivo eléctrico, detectando las variaciones en el momento que se produce el cambio de estado. Por otra parte, las firmas transitorias requieren registrar las medidas de consumo a alta frecuencia para analizar la forma y los ciclos de la señal obtenida durante el intervalo de tiempo que se estudia. Finalmente, las firmas intrusivas requieren intervención en la red eléctrica, existiendo dos sub categorías, firmas intrusivas físicamente (*Pyhisically Intrusive Signatures*) y firmas intrusivas eléctricamente (*Electrically Intrusive Signatures*). Las firmas intrusivas físicamente requieren la instala-

ción de artefactos junto a los electrodomésticos estudiados, que se encargan de medir los pulsos generados por los electrodomésticos durante sus activaciones. Por otra parte, las firmas intrusivas eléctricamente se obtienen a partir de un mecanismo que funciona de manera similar a un sonar. En este caso, se envía un pulso hacia toda la red para estudiar la distorsión de la onda generada obteniendo información sobre los electrodomésticos encendidos.

4.2. Dissagregation of Domestic Smart Meter Energy Data

La investigación de Kelly y Knottenbelt [7] consistió en la aplicación de redes neuronales al problema de desagregación. Los autores plantearon tres arquitecturas que toman como entrada una medida de consumo agregado y son entrenadas de manera individual para cada electrodoméstico. Las tres variantes desarrolladas fueron una red LSTM, una red *Denoising Autoencoder* y una red que denominaron de rectángulos.

En la red recurrente LSTM los autores utilizaron seis capas con la siguiente arquitectura:

- Capa de entrada determinada por el tamaño de ventana utilizado
- Capa convolución 1D con función de activación lineal
- Capa bidireccional LSTM con 128 neuronas
- Capa bidireccional LSTM con 256 neuronas
- Capa densa con 128 neuronas y función de activación TanH
- Capa de salida de una neurona con función de activación lineal

Los *autoencoders* son un tipo de redes neuronales particularmente bueno para extraer características. Reducen las dimensiones de la entrada mediante capas codificadoras que compactan la información y luego tratan de reconstruirla en las capas decodificadoras. Un *denoising autoencoder* es una variante donde se introduce un ruido en la señal de entrada y se busca obtener la señal original como salida. Kelly y Knottenbelt propusieron utilizar el consumo agregado como la señal de entrada distorsionada y el consumo real de cada electrodoméstico como el objetivo de predicción. La arquitectura propuesta fue:

- Capa de entrada determinada por el tamaño de ventana utilizado
- Capa convolución 1D con función de activación lineal
- Capa densa con cantidad de neuronas N = *(tamaño_secuencia - 3) * 8* y función de activación ReLU
- Capa densa con 128 neuronas y función de activación ReLU
- Capa con cantidad de neuronas N = *(tamaño_secuencia - 3) * 8* y función de activación ReLU
- Capa convolución 1D con función de activación lineal

Finalmente, la red de rectángulos debe su nombre a que busca predecir tres valores: el momento de encendido, el momento de apagado y el promedio de consumo en el intervalo. Para obtener estos valores los autores plantearon la siguiente arquitectura de red:

- Capa de entrada determinada por el tamaño de ventana utilizado
- Capa convolución 1D con función de activación lineal
- Capa convolución 1D con función de activación lineal
- Capa densa con 4096 neuronas y función de activación ReLU
- Capa densa con 3072 neuronas y función de activación ReLU
- Capa densa con 2048 neuronas y función de activación ReLU
- Capa densa con 512 neuronas y función de activación ReLU
- Capa densa con 3 neuronas y función de activación lineal

El conjunto de datos utilizado para el entrenamiento fue el repositorio de UK-DALE. Este repositorio surgió de un trabajo previo de Kelly que consistió en la recopilación de registros de consumo agregado y desagregado de cinco casas del Reino Unido. Este repositorio es descrito en profundidad en el Capítulo 6, ya que fue el utilizado en la evaluación experimental de los algoritmos implementados en el proyecto.
Por otra parte, Kelly y Knottenbelt utilizaron la biblioteca *NILMTK*, que fue desarrollada por el primero de estos en conjunto a otros investigadores en el año 2014. Esta biblioteca permite entre otras cosas generar datos sintéticos de consumo. En los resultados obtenidos por Kelly y Knottenbelt las redes neuronales que se entrenaron con datos reales y sintéticos, demostraron mejor capacidad de predicción en la generalización a hogares no vistos durante el entrenamiento.

En la comparación de las tres redes se destacaron la variante de rectángulos y el *autoencoder*. Por su parte, la red LSTM quedó por atrás en la mayoría de los electrodomésticos. Kelly y Knottenbelt concluyeron que la red LSTM tiene un buen desempeño en aquellos electrodomésticos con solo dos estados de ejecución. En en el resto de los electrodomésticos, sus malos resultados se pueden atribuir a que las características que determinan los estados suceden cada largos intervalos y quedan fuera de la memoria de la red.

4.3. Resumen

A modo de resumen el trabajo de Hart describe conceptos fundamentales del problema de desagregación energética y establece un marco teórico de trabajo sobre el mismo. En particular este proyecto se enfocó en su modelo ON/OFF con un conjunto de datos recopilados mediante la intrusión física de la red eléctrica.
La tesis de Kelly, por otra parte, tiene un enfoque práctico que aborda el uso de redes neuronales para la caracterización energética. Si bien no se utilizaron los mismos modelos que presentaron Kelly y Knottebelt, sirvieron como base para el desarrollo de los clasificadores que se presentan en el Capítulo 5.

Capítulo 5

Implementación

En este capítulo se describen los distintos clasificadores implementados en el proyecto. En la Sección 5.1 se introducen las bibliotecas utilizadas para el desarrollo, mientras que en la Sección 5.2 se presentan los clasificadores.

5.1. Bibliotecas

La implementación de los clasificadores se llevó a cabo con el lenguaje de programación *Python*, en su versión *3.6*.
En la Subsección 5.1.1 se presenta la biblioteca *NumPy* que provee estructuras de datos eficientes para trabajar sobre grandes cantidades de información. En la Subsección 5.1.2 se presenta *Pandas*, una biblioteca de alto nivel que utiliza *Numpy* y también provee estructuras para la manipulación de datos. En la Subsección 5.1.3 se describe *Scikit-Learn*, biblioteca de aprendizaje automático que ofrece implementaciones de los clasificadores *Naive Bayes* y *K Nearest Neighbors*, entre otros. En la Subsección 5.1.4 se presenta *Tensorflow*, una biblioteca de cálculo basada en grafos, mientras que en la Subsección 5.1.5 se introduce *Keras*, biblioteca que utiliza las funcionalidades provistas por *Tensorflow* para implementar diferentes arquitecturas de redes neuronales.

5.1.1. Numpy

NumPy [8] es un módulo para *Python* y su nombre es un acrónimo de *Numeric Python*. Se encuentra implementado en *C* por lo cual brinda una gran velocidad de ejecución de las funciones numéricas y matemáticas que provee.
A través del módulo *NumPy* están a disposición estructuras de datos tales

como matrices y arreglos multidimensionales que permiten trabajar con los datos de manera eficiente .
NumPy se basa en dos módulos de *Python* anteriores. Uno de estos es *Numeric*, el cual es un módulo de *Python* utilizado para computación numérica de alto rendimiento, pero que hoy en día se encuentra obsoleto. Otro predecesor de *NumPy* es *Numarray*, que es una reescritura de *Numeric* pero también se encuentra obsoleto. *NumPy* es una fusión de los anteriores, es decir, se basa en el código de *Numeric* y en las características de *Numarray*.

Las ventajas de usar *"Core Python"*, es decir, *Python* sin ningún módulo de extensión son: disponer de objetos numéricos de alto nivel como enteros y punto flotante. Utilizar contenedores como listas y diccionarios con métodos de inserción, eliminación, y unión eficientes.

Las ventajas de incorporar *Numpy* con *Python* es utilizar un paradigma orientado a arreglos multidimensionales, diseñado para computación científica de forma eficiente.

Por la complejidad que implica trabar con los grandes volúmenes de datos presentes en el proyecto, se optó por utilizar *Numpy* y las poderosas estructuras de datos que ofrece.

5.1.2. Pandas

Pandas [9] es una biblioteca de código abierto de *Python*, que proporciona herramientas de análisis y manipulación de datos de alto rendimiento. En el procesamiento y análisis de datos, independientemente del origen de estos, se deben atravesar cinco etapas: cargar, preparar, manipular, modelar y analizar. *Pandas* ofrece herramientas para tratar con datos y es utilizada en una amplia variedad de campos: dominios académicos y comerciales, finanzas, economía, estadísticas, análisis y más. Entre sus características principales están: proveer herramientas para cargar datos en memoria desde diferentes formatos de archivo. Brindar estructuras de datos manipulables que permitan eliminar o insertar columnas eficientemente. Ofrecer un alto rendimiento en intersección de datos.

5.1.3. Scikit Learn

Scikit-learn [10] es una biblioteca de código abierto para aprendizaje automático desarrollada para el lenguaje *Python*, que puede interoperar con las bibliotecas *NumPy*, *SciPy* y *MatPlotlib*.

Scikit-learn posee una amplia variedad de algoritmos de aprendizaje, tanto supervisados como no supervisados. En la primer categoría se pueden encontrar implementaciones de máquinas de soporte vectorial *(Support Vector Machine, SVM)*, árboles de decisión, *Naive Bayes*, *K-Nearest Neighbors*, algoritmos de regresión y redes neuronales. Por otro lado, en los métodos no supervisados se encuentran variantes de *Clustering*.

Scikit-learn ofrece funciones y procedimientos, dentro de sus distintos módulos, que son útiles para trabajar el manejo de características, la selección y validación de modelos, la detección de valores atípicos (*outliers*) y el preprocesamiento de conjuntos de datos.
Scikit-learn es destacada por sus usuarios debido a su simpleza y claridad, además cuenta con una basta documentación y tutoriales con código de ejemplo.

5.1.4. Tensorflow

TensorFlow [11] es una biblioteca de código abierto para computación numérica desarrollada por la empresa *Google*. Es utilizada para investigación en los campos de aprendizaje automático y aprendizaje profundo. *TensorFlow* es multiplataforma y puede ejecutar en sistemas operativos de computadoras personales y en plataformas móviles. Su interfaz de programación de aplicaciones (*API*) ha sido implementada en diversos lenguajes como *Python*, *Java*, *Go* y *Rust*.
TensorFlow sigue un modelo de ejecución basado en grafos. Los nodos representan operaciones matemáticas y las aristas representan arreglos multidimensionales de datos, que reciben el nombre de tensores. En el funcionamiento de esta biblioteca primero se define el grafo que representa el modelo de aprendizaje automático y luego se ejecuta.
TensorFlow es actualmente la biblioteca para aprendizaje automático con mayor cantidad de usuarios y cuenta con una amplia comunidad formada por

desarrolladores, empresarios e investigadores que la respaldan.

5.1.5. Keras

Keras [12] es una biblioteca de código abierto para aprendizaje profundo desarrollada sobre el lenguaje *Python*. Es una interfaz de programación de aplicaciones (*API*) de alto nivel para redes neuronales que puede trabajar sobre *Tensorflow*, *Theano* [13] y *CNTK*.
Keras fue creada bajo los siguientes principios:

- Amigable con el usuario: el principal objetivo es brindar una buena experiencia al usuario mediante una *API* funcional y consistente, con una curva de aprendizaje no muy pronunciada y comentarios de errores claros y concisos.

- Modularidad: los modelos de *Keras* están compuestos por módulos completamente independientes y configurables que pueden unirse entre si teniendo en cuenta muy pocas restricciones. Los módulos más destacables son las funciones de costo, los optimizadores, las capas neuronales y las funciones de activación.

- Fácil de extender: agregar nuevos módulos, extender clases y funciones existentes, es un proceso sencillo.

- Integración con Python: Los modelos se declaran en lenguaje *Python* que es bien estructurado y fácil de depurar.

La estructura de datos central de *Keras* es el modelo, que se representa como una pila de capas que terminan definiendo la arquitectura de una red neuronal. Esta biblioteca provee soporte para poder trabajar con múltiples variantes de redes neuronales incluyendo simples, convolucionales y recurrentes.

5.2. Clasificadores implementados para el problema de caracterización energética

En esta sección se describen detalles de los clasificadores implementados para atacar el problema de la caracterización energética.

5.2.1. Naive Bayes

Naive Bayes es uno de los algoritmos de clasificación más sencillo y rápido. Para la predicción de una clase desconocida utiliza el teorema de probabilidad de *Bayes*. [6]

La biblioteca *Scikit-Learn* implementa distintas variantes de Naive Bayes, entre ellas están:

- Bernoulli: en esta alternativa las características son variables booleanas independientes. El modelo *Bernoulli* es comúnmente usado para las tareas de clasificación de documentos, donde interesa la ocurrencia de términos binarios en lugar de la frecuencia de los términos.

- Multinomial: es usado cuando se tiene un número finito de clases donde no importa el orden. Con un modelo de evento multinomial, las muestras (vectores de características) representan las frecuencias con las que ciertos eventos han sido generados por una distribución multinomial (p_1, ..., p_n) donde p_i es la probabilidad de que ocurra el evento i.

- Gaussiano: el algoritmo *gaussiano* se utiliza en los casos que se debe trabajar con características continuas.

Dada la naturaleza de los datos y su composición, se decidió utilizar el algoritmo *gaussiano*. Para la implementación del clasificador se utilizaron dos métodos provistos por *Scikit-Learn* en *Python*:

- *fit(self, X, y)*: método de ajuste que recibe el conjunto de entrenamiento.
- *predict(self, X)*: método de predicción que recibe el conjunto de testeo como parámetro. Los resultados retornados por este método son los que se comparan con los valores reales.

El pseudocódigo del algoritmo utilizado para la implementación del clasificador Naive Bayes se presenta en el Algoritmo 1

Algoritmo 1 Naive Bayes

```
1: [Definicion de variables]
2:
3: path_to_application_train
4: # Path al archivo que contiene los vectores de entrenamiento
   con sus características correspondientes
5:
6: path_to_application_test
7: # Path al archivo que contiene los vectores de testeo con sus
   características correspondientes
8:
9: [Generación de vectores de entrada]
10:
11: file1 = open(path_to_application_train)
12: dataset_json = file1.read()
13: dataset_dic = json.loads(dataset_json)
14: x_train = dataset_dic["features"]
15: y_train = dataset_dic["activations"]
16:
17: file2 = open(path_to_application_test)
18: dataset_json2 = file2.read()
19: dataset_dic2 = json.loads(dataset_json2)
20: x_test = dataset_dic2["features"]
21: y_test = dataset_dic2["activations"]
22:
23: [Inicializacion del clasificador]
24:
25: gnb = GaussianNB()
26:
27: [Entrenamiento del clasificador]
28:
29: gnb.fit(np.array(x_train), np.array(y_train))
30:
31: y_pred = gnb.predict(np.array(x_test))
```

5.2.2. K Nearest Neighbors

Al igual que para *Naive Bayes*, para el desarrollo de *K Nearest Neighbors Classifier* se utilizó una de las implementaciones provistas por *Scikit-learn*, que ofrece las siguientes variantes para este clasificador:

- *K Neighbors Classifier*: implementa el aprendizaje basado en los k vecinos más cercanos de cada punto de consulta, donde k es un valor entero especificado por el usuario.

- *Radius Neighbors Classifier*: implementa el aprendizaje basado en el número de vecinos dentro de un radio fijo r de cada punto de entrenamiento, donde r es un valor de punto flotante especificado por el usuario.

La clasificación de k vecinos en *K Neighbors Classifier* es la técnica más utilizada y la que se eligió para la implementación de este clasificador. Dentro del conjunto de parámetros que acepta el constructor de la clase se decidió centrar el estudio en los dos que se entendieron como los más relevantes:

- n_neighbors: establece la cantidad de vecinos cercanos con los que se compara la clase del elemento a clasificar. Por defecto, el número está establecido en cinco vecinos.

- weights: indica como afecta en la clasificación la distancia de los vecinos al elemento a clasificar. El valor por defecto (*uniform*) indica que todos los vecinos tienen la misma importancia en la predicción. Sin embargo, en algunas circunstancias, es mejor darle mayor importancia a los vecinos más cercanos al punto a clasificar. Esto se puede lograr asignándole a este parámetro el valor de *distance*.

Al igual que en el clasificador anterior, se utilizaron los métodos *fit* y *predict* de *Scikit-learn*. Los datos fueron divididos en un conjunto de entrenamiento compuesto por el 80 % del total, en un conjunto validación del 10 % y uno de testeo con el 10 % restante. Como se mencionó en la sección 3.1, el conjunto de validación es utilizado para estudiar distintas configuraciones de los parámetros disponibles hasta encontrar la óptima. En este proyecto se trabajó sobre *n_neighbors* y *weights*.
El pseudocódigo del algoritmo utilizado para la implementación del clasificador K Nearest Neighbors se presenta en el Algoritmo 2.

Algoritmo 2 K Nearest Neighbors

```
1: [Definición de variables]
2:
3: path_to_application_train
4: #Path al archivo que contiene los vectores de entrenamiento
   con sus características correspondientes
5:
6: path_to_application_test
7: #Path al archivo que contiene los vectores de testeo con
   sus características correspondientes del electrodoméstico
   estudiado
8:
9: [Generación de vectores de entrada]
10:
11: file1 = open(path_to_application_train)
12: dataset_json = file1.read()
13: dataset_dic = json.loads(dataset_json)
14: x_train = dataset_dic["features"]
15: y_train = dataset_dic["activations"]
16:
17: file2 = open(path_to_application_test)
18: dataset_json2 = file2.read()
19: dataset_dic2 = json.loads(dataset_json2)
20: x_test_data = dataset_dic2["features"]
21: y_test_data = dataset_dic2["activations"]
22:
23: [Generación conjunto de testeo]
24:
25: x_test = x_test_data[len(x_test_data)//2:]
26: y_test = y_test_data[len(x_test_data)//2:]
27:
28: [Inicialización del clasificador]
29:
30: neigh = KNeighborsClassifier(n_neighbors, weights)
31:
32: [Entrenamiento del clasificador]
33:
34: neigh.fit(np.array(x_train), np.array(y_train))
35:
36: [Predicicón del clasificador]
37:
38: y_pred_test = neigh.predict(np.array(x_test))
```

5.2.3. Perceptrón multicapa

El perceptrón multicapa fue el primer acercamiento a las redes neuronales que se abordó. Para su implementación se utilizó la biblioteca *Tensorflow* presentada anteriormente.
La arquitectura del perceptrón multicapa consistió en una capa de entrada con una neurona para cada *feature*, una capa oculta con cinco neuronas y una capa de salida con dos neuronas, una asociada a la clase encendido y la otra a la clase apagado. Se optó por la función *Sigmoid* como activación y no se utilizó *bias* en las neuronas de la red. Los pesos se ajustaron mediante descenso por gradiente con una tasa de aprendizaje de 0.005. Debido a su extenso tamaño, el conjunto de entrenamiento debió ser procesado en lotes de 2000 registros, entrenándose la red durante diez épocas con cada uno. Los diferentes dominios presentados por las características ingresadas a la red implicaron que fuese necesario normalizar cada uno de ellos al intervalo [0,1]. Finalmente se utilizó el error cuadrático para evaluar las predicciones realizadas por la red.
La configuración paramétrica final se obtuvo a través de una etapa de evaluación experimental, sin embargo para este modelo de red la variación de los parámetros no afectó los resultados. El comportamiento anterior sobre la variación de parámetros y su no alteración en los resultados se explica en la siguiente sección.
Si se supone un tamaño de lote S donde cada vector de entrada tiene C características, la salida de la red se puede definir mediante la Ecuación 5.1, donde X_{SxC} es la matriz de entrada. $W1_{Cx5}$ es la matriz asociada a los pesos sinápticos entre las neuronas de la capa de entrada y la capa oculta. $W2_{5x2}$ es la matriz asociada a los pesos sinápticos entre las neuronas de la capa oculta y la capa de salida. Y_{Sx2} es la matriz de salida.

$$Y = Sigmoid(Sigmoid(X.W1).W2) \tag{5.1}$$

5.2.4. Long Short Term Memory

En esta subsección se explica cómo se implementaron este tipo de redes neuronales recurrentes para la resolución de la variante binaria del problema de caracterización. Se describe el procesamiento de la información de los conjuntos de datos trabajados, la selección de las arquitecturas utilizadas, las bibliotecas empleadas y sus respectivas configuraciones paramétricas, entre los aspectos

más destacables.

Aspectos comunes a todas las variantes LSTM

Las implementaciones de las redes *LSTM* fueron realizadas mediante el uso de la biblioteca *Keras*. Este tipo de redes necesita que los datos que se procesan sean enviados de la siguiente forma: ***[samples, timesteps, features]***, donde *samples* hace referencia a la cantidad de ejemplos o instancias de entrenamiento, *timesteps* a el largo de la secuencia de observaciones anteriores o pasos hacia atrás que se le asocia a una salida, y *features* representa la cantidad de características de cada uno de los objetos a clasificar.

Se deben agrupar los vectores de entrada en conjuntos de tamaño igual al valor indicado por *timesteps*. Asímismo, se debe mapear dichas agrupaciones a una salida determinada. Con el fin de no perder ningún tipo de información de entrada, el método utilizado para formar los grupos se describe a continuación. Dado un entero N (*timesteps*) que representa la cantidad elementos que conforman la ventana de memoria, se forma un grupo con los primeros N vectores de la entrada y se le asocia la salida del vector $N + 1$. El próximo grupo está conformado por el segundo vector hasta el vector $N + 1$ y se le asocia la salida del elemento $N + 2$, y así sucesivamente. Conceptualmente, esto significa que la red tomará la entrada de los N instantes anteriores para determinar la salida actual. Se tomó la decisión de descartar los primeros N vectores ya que no se cuenta con la información necesaria para conformar su ventana. El pseudocódigo de los programas que implementan el procedimiento antes descrito se muestra en los Algoritmos 3 y 4.

Algoritmo 3 Generación de vectores de entrada LSTM

```
1: es_multiplo ← False
2: while not es_multiplo do
3:     if length(data) mód time_steps == 0 then
4:         es_multiplo ← True
5:     else
6:         delete_one_element(data)
7:     end if
8: end while
9: secuence_vector ← []
10: for index, value ∈ enumerate(data) do
11:     aux ← []
12:     if length(data) < time_steps + index then
13:         break
14:     end if
15:     for i ∈ range(time_steps) do
16:         aux. append(data[index + i])
17:     end for
18:     secuence_vector. append(aux)
19: end for
20: return secuence_vector
```

Algoritmo 4 Generación de etiquetas de vectores LSTM

```
1: es_multiplo ← False
2: while not es_multiplo do
3:     if length(data) mód time_steps == 0 then
4:         es_multiplo ← True
5:     else
6:         delete_one_element(data)
7:     end if
8: end while
9: for i ∈ range(time_steps) do
10:     delete_first_element(data)
11: end for
12: return data
```

La cantidad de capas que definen al modelo y sus parámetros (tipo, cantidad de neuronas, función de activación, etc), fueron evaluados con diferentes combinaciones de valores con el fin de encontrar la configuración que determina los mejores resultados.

Keras permite definir el algoritmo de optimización así como también la función de costo con la que se va a entrenar la red y evaluar las predicciones por ella realizadas. Para todos los casos que se exponen en la siguiente sección se utilizó *Adam* como algoritmo de optimización ya que es una mejora al descenso por gradiente tradicional tal como se explicó en el capítulo 3 . Se optó por *Binary Cross Entropy* como función de costo debido a que se está trabajando con una salida binaria. Así mismo, se definió en 15 la cantidad de *epochs* de entrenamiento, ya que en la evaluación experimental se constató que la función de costo se estabilizaba con valores mayores a este.

Por otro lado, se hizo uso también de la biblioteca *Scikit-learn* para resolver dos inconvenientes. El primero tiene que ver con la normalización de los datos de entrada, se llevó cada característica al intervalo [0,1] mediante la función *MinMaxScaler* del módulo *preprocessing* ofrecido por la biblioteca. El segundo inconveniente se refiere al problema del desbalanceo de instancias dentro de los datos de entrenamiento. Las instancias que pertenecen a la clase apagada superan ampliamente a las de la clase prendida. Con el fin de mitigar este contratiempo, se calculó la proporción entre la cantidad de instancias de cada clase. Se aumentó la relevancia en la función de costo de las instancias de la clase débil según la proporción calculada durante el entrenamiento de la red neuronal en cuestión. Para esta modificación de la relevacia se utilizó el parámetro *class_weight* del método *fit* de *Keras* con la proporción antes mencionada.

Arquitecturas

Se construyeron varias arquitecturas con el objetivo de poder encontrar el mejor clasificador para cada uno de los distintos electrodomésticos considerados en el estudio. Las arquitecturas trabajadas fueron:

- LSTM simple: esta variante está conformada por una sola capa de *LSTM* seguida de una capa totalmente conectada (densa). Para la primer capa, la cantidad de neuronas se estableció en 128, se escogió como función de activación *ReLU*. La segunda capa cuenta con una sola neurona y una función de activación *sigmoid*.

- LSTM apilado: en este caso se construyó una estructura compuesta por tres capas *LSTM* y una capa totalmente conectada. La primer capa posee 128 neuronas y su función de activación es *TanH* (tangente hiperbólica). La segunda y la tercera capas comparten la misma función de activación que la primera, pero se componen de 256 y 128 neuronas respectivamente. El último componente cuenta con una neurona y una función de activación de tipo *sigmoid*.

- LSTM apilado bidireccional: mantiene la base de estructura del caso anterior pero en esta oportunidad las primeras dos capas no son simples *LSTM*, sino que son *LSTM* bidireccionales. Asímismo, se comparten las cantidades de neuronas y funciones de activación de las capas.

- CNN-LSTM: finalmente, la última configuración mezcla dos tipos de redes neuronales presentadas anteriormente y busca aprovechar las cualidades de ambas. En primer lugar intenta extraer las características de los datos de entrada apoyándose en *CNN* para luego poder trabajar con problemas de secuencia mediante el uso de *LSTM*. Se compone de una capa convolucional de 64 filtros y función de activación *ReLU*, seguido de una capa *MaxPooling1D* con un tamaño de ventana igual a dos y por último una capa de tipo *Flatten*. A continuación se agregan dos capas más, la primera una *LSTM* con 128 neuronas y función de activación *ReLU* y para culminar una capa completamente conectada compuesta por una neurona y *sigmoid* como función de activación.

Keras cuenta con implementaciones reducidas de ciertas capas como *LSTM* con el fin de acelerar los procesos de entrenamiento. Estas variantes fueron desarrolladas a su vez para sacar provecho del uso de unidades de procesamiento gráfico (*Graphics Processing Units, GPUs*). Las capas reducidas disminuyeron el tiempo de entrenamiento en un 75 % aproximadamente, bajando de un promedio de 30 horas por entrenamiento, a uno de 8 horas al aprovechar el dispositivo de video. Sin embargo, estas capas presentan la peculiaridad de no permitir cambiar su función de activación, siendo esta la razón de por la cual todas ellas comparten la misma función de activación.

Capítulo 6

Evaluación experimental

En este capítulo se presenta la evaluación experimental de los distintos algoritmos desarrollados para resolver el problema de desagregación de consumo eléctrico en hogares. En la Sección 6.1 se detallan los repositorios de datos sobre los cuales se ejecutaron los algoritmos. El apartado 6.2 describe los procedimientos que se llevaron a cabo sobre los datos para extraer características y construir las diferentes entradas que se ingresaron en los clasificadores. La Sección 6.3 explica los entornos en los cuales fueron ejecutados cada uno de los algoritmos. Por su parte, el apartado 6.4 expone las distintas métricas de desempeño utilizadas para evaluar los resultados. La Sección 6.5 ilustra los resultados obtenidos de cada método sobre los electrodomésticos en estudio. Finalmente la Sección 6.6 presenta el desempeño de los clasificadores frente a datos de hogares con conductas de consumo eléctrico no conocidas.

6.1. Repositorios de datos

Al momento de realizar esta investigación se evaluaron los contenidos de dos repositorios de datos, *The Reference Energy Disaggregation Data Set* (REDD) [14] y *UK Domestic Appliance Level Electricity* (UK-DALE) [15]. Se escogió trabajar con el último de estos debido a que presenta una mayor cantidad de datos y mayor variedad de electrodomésticos evaluados. A partir de estos datos, se generaron los conjuntos de entrenamiento, validación y testeo utilizados por los distintos clasificadores implementados.

6.1.1. UK-DALE

El conjunto de datos *UK-DALE* contiene información sobre consumo eléctrico de varios electrodomésticos en cinco hogares del Reino Unido en el período entre los años 2012 y 2017. UK-DALE posee lecturas etiquetadas de consumo agregado y desagregado de los electrodomésticos presentes en los hogares. Los electrodomésticos presentes en cada uno de los hogares no son necesariamente los mismos.
La frecuencia de muestreo del consumo agregado y de los particulares de cada electrodoméstico es de $1/6Hz$. La casa que presenta la mayor cantidad de electrodomésticos muestreados es la número uno con 52, el resto oscila entre 4 y 24. La información de cada electrodoméstico medido es guardada en archivos de texto plano, donde se registran dos columnas. La primer columna refiere al instante en el que fue tomada la medida y se representa como una marca de tiempo (*timestamp*) de tipo *UNIX*. La segunda columna, indica el consumo del electrodoméstico en ese momento y su unidad es el *Watt*.
El consumo del electrodoméstico no es la única información con la que cuenta este conjunto de datos. Para los electrodomésticos de las casas uno, dos y cinco también se tomaron medidas del consumo activo y consumo aparente cada un segundo que se almacenaron en archivos de extensión *.CSV*.
Se seleccionaron cinco electrodomésticos del hogar uno sobre los cuales trabajar: **heladera, jarra eléctrica, microondas, lavavajillas y lavarropas**. El criterio de selección estuvo basado en dos aspectos: el impacto que tiene el electrodoméstico sobre el cosumo agregado y la duración de uso del mismo. El lavarropas y el lavavajillas presentan ciclos de funcionamiento prolongados. En el microondas y la jarra eléctrica los períodos de encendido son cortos, sin embargo, el consumo empleado es alto. Finalmente, la heladera tiene la particularidad de activarse sin la intervención del factor humano. Los datos de cada electrodoméstico se fraccionaron en conjuntos de entrenamiento, validación y testeo siguiendo la proporción 80 %, 10 % y 10 % respectivamente.

6.2. Definición de los escenarios de evaluación

Los métodos de aprendizaje automático con los que se trabajó en el proyecto de grado son miembros de la familia de los métodos supervisados, por lo cual necesitan de datos etiquetados para poder reafirmar o corregir los resul-

tados durante el entrenamiento.
Como ya fue explicitado anteriormente, esta investigación tuvo como pilar fundamental los datos provistos por el repositorio de datos *UK-DALE*. Sobre la información que el conjunto de datos provee, se idearon distintos escenarios de evaluación con el objetivo de encontrar el conjunto de características que mejor identifica al comportamiento de los electrodomésticos. Las características consideradas en la etapa de evaluación experimental fueron las siguientes:

- CONSUMO_AGREGADO: es un valor real que representa el consumo agregado del hogar en el instante de tiempo anterior.

- ENCENDIDO_ANTERIOR: es una característica booleana que indica si el electrodoméstico se encontraba encendido en el instante de tiempo anterior. El valor 1 se corresponde con el encendido del electrodoméstico y el valor 0 con el apagado.

- ESTADO_ANTERIOR: es una característica que extiende ENCENDIDO_ANTERIOR para la variante del problema con múltiples niveles de consumo. Es un valor entero que corresponde al identificador del estado en el cual se encontraba el electrodoméstico en el instante de tiempo anterior.

- TIEMPO_APAGADO: es una característica que toma valores enteros mayores o iguales a cero e indica el tiempo en segundos transcurridos desde que se registró el último encendido del electrodoméstico.

- TIEMPO_ENCENDIDO: es una característica que toma valores enteros mayores o iguales a cero e indica el tiempo en segundos transcurridos desde que se registró el último apagado del electrodoméstico.

- FIN_SEMANA: es una característica booleana que indica si el instante en el que se realizó la medición de consumo corresponde a un día de semana o a fin de semana. Toma el valor 0 para los días comprendidos entre lunes y viernes. Toma el valor 1 para los días sábados y domingos.

- HORA_DIA: es una característica que indica la hora del día en la que se

realizó la medición de consumo. Toma valores enteros entre 0 y 23.

- VARIACION_CONSUMO: es una característica que toma valores reales e indica la variación del consumo agregado del hogar entre los dos instantes de tiempo anteriores.

- FRANJA: es una característica que surge a partir de VARIACION_CONSUMO y busca agrupar aquellas variaciones similares relacionadas con el mismo comportamiento de un electrodoméstico. Las franjas son intervalos de consumo cuyo ancho es el valor de consumo standby. El valor de esta característica está dado por la siguiente expresión:

$$FRANJA = round(VARIACION\_CONSUMO/consumo\_standby)$$

Se crearon un total de nueve escenarios distintos a lo largo del proyecto, contabilizando los diseñados para la variante binaria del problema y para la variante de múltiples niveles de consumo. Los escenarios se describen en las siguientes subsecciones mientras que la Tabla 6.1 presenta un resumen de las características incluidas en cada escenario.

6.2.1. Variante Binaria

En cinco de los siete escenarios creados para trabajar con la variante binaria se utilizó la información de encendido anterior del electrodoméstico como una característica de entrada. La decisión de su valor se basó en la siguiente lógica: un electrodoméstico se clasifica encendido si el consumo en ese momento sobrepasa un cierto valor constante denominado *standby*.
Cada uno de los electrodomésticos en cuestión posee su propio valor de consumo *standby*: en la jarra eléctrica y el lavarropas este valor se definió en 50 Watt. Para el microondas se estableció en 30 Watt, para el lavavajillas se definió en 20 Watt y para la heladera en 60 Watt.

Escenario 1

Para el escenario 1 se consideran ciertas características que tienen en cuenta el comportamiento de los electrodomésticos en tiempos anteriores. Se mantiene información acumulada sobre los encendidos y apagados ocurridos hasta el

momento. Se considera el estado anterior de los electrodomésticos y el tiempo que lleva apagado. También se agregan características adicionales como el consumo total del hogar (en el instante de tiempo) y el día de la semana al que pertenece ese consumo (se definen dos clases, una correspondiente a los días entre lunes y viernes, y una segunda clase para los sábados y domingos). Las características que se utilizan en este escenario son:

- CONSUMO_AGREGADO
- ENCENDIDO_ANTERIOR
- TIEMPO_APAGADO
- FIN_SEMANA

Escenario 2

El escenario 2 presenta las siguientes características:

- FIN_SEMANA
- CONSUMO_AGREGADO

El escenario 1 asume que se tiene conocimiento previo sobre el comportamiento del electrodoméstico, o que se pueden utilizar las salidas del propio algoritmo en instantes previos como entrada para las predicciones siguientes. Por otra parte, el escenario 2 considera una situación en la que no se dispone de información acumulada sobre el comportamiento del electrodoméstico.

Escenario 3

Para las características de los escenarios 1 y 2, se consideran los registros de los electrodomésticos para los cuales existe una medida de consumo agregado en los registros del hogar con el mismo *timestamp*. Por tanto, si no existe una medida coincidente se descarta ese registro, desperdiciando de ese modo, aproximadamente el 90 % de las mediciones de cada electrodoméstico.
El escenario 3 incorpora la discretización del tiempo en intervalos de tamaño fijo t, para evitar descartar registros. En el proyecto se decidió trabajar con $t = 30$ segundos, ya que los consumos de los electrodomésticos del conjunto de datos *UK-DALE* se muestrean cada seis segundos. De este modo, se obtienen intervalos de largo múltiplo de la frecuencia de muestreo.

El primer intervalo comienza en la primer medida de *timestamp* de las entradas de cada electrodoméstico y finaliza t segundos más tarde. Los restantes intervalos se construyen de manera iterativa repitiendo el proceso, esta vez tomando como inicio el final del intervalo anterior. Una vez definido un intervalo, se buscan las lecturas de consumo que corresponden al rango de tiempo en consideración.
Al mismo tiempo, se consideró una característica adicional a las utilizadas anteriormente, la hora del día. Finalmente, las características utilizadas en el escenario 3 son:

- CONSUMO_AGREGADO: Como en este escenario se consideran intervalos de tiempo, esta característica representa el consumo agregado promedio del hogar en el intervalo. Si el electrodoméstico se encuentra encendido basta con hacer el promedio de los consumos agregados en ese intervalo. En caso de considerarse al electrodoméstico apagado, se deben restar los consumos del electrodoméstico en ese intervalo y luego hacer el promedio del consumo agregado del hogar.

- ENCENDIDO_ANTERIOR: Como en este escenario se consideran intervalos de tiempo, para determinar si el electrodoméstico se encuentra encendido se contabilizan las medidas, siendo positivo si más de la mitad de estas supera el valor de consumo *standby*.

- TIEMPO_APAGADO

- FIN_SEMANA: Como en este escenario se consideran intervalos de tiempo, primer medida del intervalo para determinar el valor de esta característica.

- HORA_DIA: Como en este escenario se consideran intervalos de tiempo, se utiliza la primer medida del intervalo para determinar el valor de esta característica.

Escenario 4

Al abordar el área de redes neuronales fue necesario contar con entradas normalizadas, lo que llevó a definir un nuevo escenario. El escenario 4 mantiene la discretización del tiempo en intervalos de 30 segundos y utiliza las mismas características que el escenario 3 pero normalizadas.

Escenario 7

Tras haber experimentado con la variante de múltiples niveles de consumo, que llevó a la creación de los escenarios 5 y 6, se decidió volver al problema binario tratando de mejorar las características presentes en los vectores de entrada.
Los escenarios 1, 2, 3 y 4 tienen en cuenta el impacto del consumo de cada electrodoméstico en el agregado del hogar, pero no analizan la variación de consumo del propio electrodoméstico de manera directa. El escenario 7 se conformó al agregar al escenario 4 las características *VARIACION_CONSUMO* y *TIEMPO_ENCENDIDO*. Esta última característica se agregó para ayudar a detectar el cambio de estado de encendido a apagado, ya que la característica *TIEMPO_APAGADO* no aporta información si el electrodoméstico se encuentra encendido. Las características utilizadas son las siguientes:

- ENCENDIDO_ANTERIOR
- TIEMPO_APAGADO
- TIEMPO_ENCENDIDO
- FIN_SEMANA
- HORA_DIA
- VARIACION_CONSUMO

En los algoritmos de aprendizaje profundo se normalizaron la características de este escenario, mientras que en Naive Bayes y K Nearest Neighbours no fue así. De aquí hasta el final del documento se refiere al Escenario 7 indistintamente de la normalización de sus características.

Escenario 8

En el escenario 8 no se tuvieron en cuenta las características de *ENCENDIDO_ANTERIOR*, *TIEMPO_ENCENDIDO*, *TIEMPO_APAGADO*, considerando que podrían ser la causa de *overfitting*. Específicamente, un método cuya

salida sea igual al valor del estado del electrodoméstico en el instante anterior, tendrá métricas de desempeño altas pero será incapaz de detectar los cambios de estados que se produzcan. Por tanto, se optó por crear este escenario donde se utiliza como característica únicamente la variación de consumo.

Escenario 9

El escenario 9 es una variante del escenario 7 donde se comparten las características pero se seleccionan de manera particular los datos con los que se entrenan los clasificadores.
Con el objetivo de intentar predecir de forma más certera los cambios de estados ocurridos, se optó por generar un conjunto de entrenamiento a partir del período de tiempo con mayor cantidad de cambios de estado. Los conjuntos de validación y testeo no sufren modificaciones en su generación con respecto a los escenarios 1, 2, 3, 4 y 7.

6.2.2. Variante con mútliples niveles de consumo

En esta subsección se describen los escenarios 5 y 6 definidos para la variante del problema de desagregación con múltiples niveles de consumo.

Escenario 5

El escenario 5 es una versión análoga a los escenarios 3 y 4, donde se reinterpretan las características consideradas para adaptarse al problema de múltiples niveles de consumo. Se introducen las características *TIEMPO_ENCENDIDO* y *FRANJA*. Las características utilizadas en el escenario 5 son:

- ESTADO_ANTERIOR
- FRANJA
- TIEMPO_APAGADO
- TIEMPO_ENCENDIDO
- FIN_SEMANA
- HORA_DIA

Escenario 6

El escenario 6 es una variante del escenario 5 donde se propone utilizar únicamente la variación de consumo agregado (la característica *FRANJA*).

Tabla 6.1: Resumen características consideradas en cada escenario.

Escenario	Variante	Características
1	Binaria	CONSUMO_AGREGADO ENCENDIDO_ANTERIOR TIEMPO_APAGADO FIN_SEMANA
2	Binaria	CONSUMO_AGREGADO FIN_SEMANA
3	Binaria	CONSUMO_AGREGADO ENCENDIDO_ANTERIOR TIEMPO_APAGADO FIN_SEMANA HORA_DIA
4	Binaria	CONSUMO_AGREGADO ENCENDIDO_ANTERIOR TIEMPO_APAGADO FIN_SEMANA HORA_DIA
5	Estados	FRANJA ESTADO_ANTERIOR TIEMPO_APAGADO TIEMPO_ENCENDIDO FIN_SEMANA HORA_DIA
6	Estados	FRANJA
7	Binaria	ENCENDIDO_ANTERIOR TIEMPO_APAGADO TIEMPO_ENCENDIDO FIN_SEMANA HORA_DIA VARIACION_CONSUMO
8	Binaria	VARIACION_CONSUMO
9	Binaria	ENCENDIDO_ANTERIOR TIEMPO_APAGADO TIEMPO_ENCENDIDO FIN_SEMANA HORA_DIA VARIACION_CONSUMO

6.3. Entorno de ejecución

La Subsección 6.3.1 detalla el entorno local sobre el cual fueron ejecutados los métodos de menor demanda computacional. Por otra parte, en la Subsección 6.3.2 se describe la plataforma sobre la cual se ejecutaron los clasificadores basados en redes neuronales, cuya demanda de recursos no podía ser satisfecha por computadores personales.

6.3.1. Entorno de ejecucón local

Los algoritmos de aprendizaje automático como *Naive Bayes y K Nearest Neighbours* no necesitan de gran poder de cómputo. Las ejecuciones se realizaron de forma local en computadoras personales, cuya unidad más potente contó un procesador Intel Core i7-8550U con velocidad de 1.8GHz y 16GB de memoria RAM.

6.3.2. Entorno de ejecución remoto

Los algoritmos de aprendizaje profundo necesitan procesar una gran cantidad de datos para aprender de manera significativa. Su entrenamiento es muy costoso desde el punto de vista computacional debido a las grandes multiplicaciones matriciales que se llevan a cabo en el proceso.
Cuanto mayor es la cantidad de observaciones con las que se cuenta para entrenar una red neuronal, cuanto más capas forman parte de su estructura y cuanto mayor es el número de neuronas que componen las capas ocultas, el tiempo que demanda el entrenamiento de una red neuronal crece de forma considerable. Con el fin de poder reducir los tiempos de ejecución se decidió hacer uso de la infraestructura del Centro Nacional de Supercomputación ClusterUY.
El Centro Nacional de Supercomputación ClusterUY [16] fue creado con el fin de proveer servicios de computo científico de alto desempeño, mediante una estructura de gran poder computacional. Esta basado en una arquitectura de agregación de componentes de hardware del tipo cluster. Ostenta además, un poder de cómputo superior al de 10.000 computadoras tradicionales y es capaz de realizar complejas operaciones matemáticas y procesamiento de datos en tan solo escasos segundos.
En ClusterUY se instalaron entornos virtuales de *Python* y se configuraron las distribuciones de *Tensorflow* que permiten hacer uso de GPUs para obtener el

máximo provecho de su arquitectura de núcleos paralelos.
La evaluación experimental de los algoritmos de aprendizaje profundo fue realizada en un nodo puntual del clúster, del que se utilizaron los siguientes recursos: un procesador Intel Xeon-Gold 6138 2.10 GHz, 16GB de memoria RAM y una GPU Nvidia Tesla P100 de 12GB.

6.4. Métricas de desempeño

Existen varias métricas o indicadores que se utilizan para evaluar el rendimiento de un clasificador. Las métricas más conocidas son *accuracy*, *precision*, *recall* y *f-score*. Para entender como funcionan cada una de las métricas hace falta explicar previamente los conceptos que están presentes en sus definiciones.
Accuracy es una medida del modelo e indica el porcentaje de predicciones que fueron acertadas. Se calcula como el total de predicciones correctas sobre el total de las predicciones.
Por otro lado, las otras tres medidas son particulares para cada una de las clases que existen. La métrica *precision*, definida en la Ecuación 6.1, es la cantidad de elementos de una clase clasificados correctamente, sobre la cantidad de elementos clasificados de esa clase en total.
En la Ecuación 6.1 *TP* o *true positives* son los elementos de la clase que se clasificaron correctamente y *FP* o *false positives* son los elementos que se clasificaron como de esa clase pero finalmente fueron de otra.

$$Precision = TP/(TP + FP) \tag{6.1}$$

La métrica *recall*, definida por la Ecuación 6.2, consiste en la cantidad de elementos clasificados correctamente de una clase, sobre la cantidad de elementos de esa clase.
TP se define de la misma forma que para la métrica anterior y *FN* o *false negatives* son los elementos que se asignaron a otra clase pero finalmente pertenecían a la clase objetivo.

$$Recall = TP/(TP + FN) \tag{6.2}$$

Por último, *f-score* es una medida armónica que considera tanto la *precision* como el *recall* y se calcula según la Ecuación 6.3, donde PR es la *precision* y RC es el *recall*.

$$F - SCORE = 2 * PR * RC/(PR + RC) \tag{6.3}$$

Mediante el análisis de las métricas *precision*, *recall* y *f-score* sobre los resultados obtenidos en el conjunto de validación, se modifican las distintas configuraciones de parámetros hasta lograr un clasificador estable para el modelo de trabajo. Como último paso, se vuelven a calcular las mismas métricas sobre un nuevo conjunto de datos. Dicho conjunto recibe la denominación de conjunto de testeo y nunca debió ser procesado por el algoritmo, con el fin de evaluar el rendimiento del clasificador.

Para evaluar el desempeño de un clasificador en el problema de caracterización de consumo eléctrico, no solo basta con visualizar los resultados obtenidos en las métricas anteriores. Se debe estudiar la capacidad del clasificador para detectar los cambios de estado en cada uno de los electrodomésticos. Para determinar qué tan eficaz es un clasificador se debe analizar la relación entre los cambios de estados predichos, acertados y reales. Un clasificador óptimo es aquel que obtiene el mismo número en los tres tipos de cambios de estado.

Para obtener las métricas antes descritas se trabajó con un módulo de la biblioteca *Scikit-Learn* llamado *metrics*, que proporciona las siguientes funciones:

- **accuracy_score**: la función devuelve la *accuracy* del clasificador sobre el conjunto de predicciones. Si todo el conjunto de etiquetas predichas para una muestra coincide con el verdadero conjunto de etiquetas, entonces la *accuracy* toma el valor ideal de 1. Toma el valor 0 en el peor caso cuando no hay predicciones correctas.
  Si $y_i^{'}$ es el valor predicho para la muestra i, y y_i es el valor real de la muestra, el valor correspondiente a las predicciones correctas sobre n muestras se se define como la Ecuación 6.4. Donde $L(x, y)$ es una función que retorna 1 si $x = y$ y 0 en caso contrario

$$accuracy(y, y^{'}) = (1/n).\sum_{i=0}^{n-1} L(y_i, y_i^{'}) \tag{6.4}$$

- **precision_score**: la precisión es la capacidad del clasificador para no etiquetar como positiva una muestra que es negativa.

- **recall_score**: el recall es la capacidad del clasificador de encontrar todos las muestras positivas.

- **confusion_matrix**: esta función devuelve como resultado una matriz y sirve para evaluar la *accuracy* de un clasificador. La función recibe como entrada los conjuntos de valores predichos y reales, y puede indicarse cómo indexar la matriz a través de etiquetas mediante el parámetro opcional *labels*. La entrada *(i,j)* en una matriz de confusión es el número de observaciones que hay en la clase i, pero que se predicen pertenecientes a la clase j. A modo ilustrativo se presenta un caso de ejemplo con dos listas de animales, y una serie de valores reales y valores predichos.

  Y = [gato, hormiga, gato, gato, hormiga, ave]
  O = [hormiga, hormiga, gato, gato, hormiga, gato]

  La matriz de confusión es la ilustrada en la Tabla 6.2:

	Hormiga	Ave	Gato
Hormiga	2	0	0
Ave	0	0	1
Gato	1	0	2

Tabla 6.2: Ejemplo de matriz de confusión.

  La diagonal de la matriz da los valores de *TP*. La celda (hormiga, hormiga) representa que se clasificaron dos hormigas y efectivamente lo eran. Los *FP* y *FN* quedan representados en cada fila, la celda (gato, hormiga) muestra que se clasificó un caso como hormiga cuando en realidad era gato.

- **classification_report**. Esta función retorna un informe donde se muestran las principales métricas, incluyendo *precision*, *recall*, *f1-score*, y otras como *support*, *micro average*, *macro average*, *weighted average*. *Support* es el número de apariciones de cada clase en los datos reales. *Micro average* y *macro average* son diferentes formas de calcular las métricas principales. Por un lado *macro average* calcula la métrica de forma independiente para cada clase y luego toma el promedio (trata a todas las clases por igual), mientras que *micro average* agrega las contribuciones de todas las clases para calcular la métrica promedio. Por último, la métrica *weighted average* proporciona un promedio ponderado de *precision*, *recall* y *f1 score* donde los pesos son los valores de soporte.

  Al igual que para la función *confusion_matrix*, se presenta a continuación un ejemplo ilustrativo de como opera *classification_report*.

  Sean las listas Y y O los valores reales y predichos por un clasificador sobre tres clases *Clase 0*, *Clase 1*, *Clase 2*:

  Y = [1, 1, 1]
  O = [1, 1, 0]

  El reporte retornado por la función *classification_report* es ilustrado en la Tabla 6.3. Con el fin de esclarecer conceptos, se expone como calcular los valores *Micro Avg*, *Macro Avg* para el caso de la métrica *recall*.

	Precision	Recall	F1 score	Support
Clase 0	0.00	0.00	0.00	0
Clase 1	1.00	0.67	0.80	3
Clase 2	0.00	0.00	0.00	0
Micro Avg	1.00	0.67	0.80	3
Macro Avg	0.33	0.22	0.27	3
Weighted Avg	1.00	0.67	0.80	3

Tabla 6.3: Ejemplo de reporte de clasificación.

$TP_0 = 2$
$TP_1 = 0$

$$TP_2 = 0$$
$$FN_0 = 1$$
$$FN_1 = 0$$
$$FN_2 = 0$$

$$MicroAvg = \frac{TP_0 + TP_1 + TP_2}{TP_0 + TP_1 + TP_2 + FN_0 + FN_1 + FN_2} = 0.67$$

$$MacroAvg = \frac{Recall_{C0} + Recall_{C1} + Recall_{C2}}{3} = 0.22$$

6.5. Resultados sobre hogares vistos durante el entrenamiento

En esta sección se presentan los resultados obtenidos por cada uno de los clasificadores implementados sobre los distintos conjuntos de datos diseñados.

Debido a que los electrodomésticos se encuentran apagados la mayor parte del tiempo, los datos de evaluación están desbalanceados. La clase apagado excede en gran proporción a la o las clases restantes y provoca una mayor facilidad para clasificar las muestras que a ella pertenecen. En la variante binaria, los valores de las matrices de confusión para las clases apagado y encendido son correlativos. Si TP_e,TN_e,FP_e,FN_e, TP_a,TN_a, FP_a, FN_a son los *true positives*, *true negatives*, *false positives* y *false negatives* de las clases encendido y apagado respectivamente, se cumple:

$$TP_e = TN_a$$
$$TN_e = TP_a$$
$$FP_e = FN_a$$
$$FN_e = FP_a$$

Debido al desbalanceo de los datos y a la correlación entre clases de la variante binaria, cuando se expresan métricas que dependen de una clase de clasificación en esta sección, corresponden a la clase encendido. Los resultados para la clase apagado se pueden ver en el Apéndice 1.

6.5.1. Naive Bayes

El clasificador *Naive Bayes* fue evaluado en los escenarios 1, 2, 3 y 7. A continuación se presentan los resultados obtenidos por este clasificador en los escenarios en los cuales fue evaluado.

Resultados Escenario 1

Los tamaños de los conjuntos de entrenamiento y testeo en el escenario 1 así como el valor de consumo *standby* para cada electrodoméstico se detallan en la Tabla 6.4. La Tabla 6.7 indica la cantidad de predicciones y salidas reales, mientras que la Tabla 6.5 muestra los valores de *TP, FP, TN, FN* que se obtuvieron para cada electrodoméstico. Los resultados expuestos en las tablas antes mencionadas son la base para el cálculo de las métricas de desempeño que se presentan en la Tabla 6.6. Finalmente, en la Tabla 6.8 se muestran los cambios de estado correctamente predichos junto a los reales.

Tabla 6.4: Descripción de los datos para Naive Bayes en el escenario 1.

Electrodoméstico	ID	Standby	Tamaño Entrenamiento	Tamaño testeo
Lavavajillas	6	20	2161955	540489
Microondas	13	30	1973212	493303
Heladera	12	60	2162831	540708
Jarra eléctrica	10	50	2124576	531144
Lavarropas	5	50	2066934	516734

Tabla 6.5: Resultados para Naive Bayes en el escenario 1.

ID	TP	FP	TN	FN
6	5864	7105	517642	9878
13	879	15480	476700	244
12	220094	8253	304194	8167
10	1461	7357	522172	154
5	5170	11729	490525	9310

Tabla 6.6: Métricas de desempeño para Naive Bayes en el escenario 1.

ID	F1 Score	Accuracy	Precision	Recall
6	0.4084	0.9686	0.4522	0.3725
13	0.1005	0.9681	0.0537	0.7827
12	0.9640	0.9696	0.9639	0.9642
10	0.2800	0.9859	0.1657	0.9046
5	0.3294	0.9593	0.3059	0.3570

Tabla 6.7: Resultados predichos y reales para Naive Bayes en el escenario 1.

ID	Encendidos (P)	Apagados (P)	Encendidos (R)	Apagados (R)
6	12969	527520	15742	524747
13	16359	476944	1123	492180
12	228347	312361	228261	312447
10	8818	522326	1615	529529
5	16899	499835	14480	502254

Tabla 6.8: Resultados acertados, reales y porcentaje de aciertos para Naive Bayes en el escenario 1.

ID	Cambios estado (A)	Cambios estado (R)	Porcentaje aciertos
6	145	1339	11
13	305	632	48
12	2	16339	0
10	954	1276	75
5	235	5966	4

En este escenario el clasificador presentó los mejores resultados en lo que respecta a las métricas de desempeño (*f1-score, accuracy, precision, recall*) sobre la heladera. Sin embargo, en este electrodoméstico se obtuvo el peor porcentaje de aciertos en cambios de estado. Por otra parte, el mayor porcentaje de aciertos se obtuvo en la jarra eléctrica, con un valor del 75 %.
De manera general, en el escenario 1 *Naive Bayes* obtuvo altos valores en los registros de *accuracy*, bajos registros de *f1-score*, y fue poca su capacidad para detectar cambios de estado.

Resultados Escenario 2

En la Tabla 6.9 se muestran las dimensiones de los conjuntos de entrenamiento y testeo que se emplearon en el escenario 2. En las tablas 6.10 y 6.11 se muestran los valores de la matriz de confusión y las métricas de desempeño calculadas. La Tabla 6.12 indica la cantidad de encendidos y apagados predichos y reales. Los cambios de estado predichos y acertados se exponen en la Tabla 6.13.

Tabla 6.9: Descripción de los datos para Naive Bayes en el escenario 2.

Electrodoméstico	ID	Standby	Tamaño Entrenamiento	Tamaño testeo
Lavavajillas	6	20	2161955	540489
Microondas	13	30	1973212	493303
Heladera	12	60	2162831	540708
Jarra eléctrica	10	50	2124576	531144
Lavarropas	5	50	2066934	516734

Tabla 6.10: Resultados para Naive Bayes en el escenario 2.

ID	TP	FP	TN	FN
6	5864	7924	516823	9878
13	679	19736	472444	444
12	9343	6224	306223	218918
10	1463	7356	522173	152
5	5187	15564	486690	9293

Tabla 6.11: Métricas de desempeño para Naive Bayes en el escenario 2.

ID	F1 Score	Accuracy	Precision	Recall
6	0.3971	0.9671	0.4253	0.3725
13	0.0631	0.9591	0.0333	0.6046
12	0.0765	0.5836	0.6002	0.0409
10	0.2804	0.9859	0.1659	0.9059
5	0.2944	0.9519	0.2500	0.3582

Tabla 6.12: Resultados predichos y reales para Naive Bayes en el escenario 2.

ID	Encendidos (P)	Apagados (P)	Encendidos (R)	Apagados (R)
6	13788	526701	15742	524747
13	20415	472888	1123	492180
12	15567	525141	228261	312447
10	8819	522325	1615	529529
5	20751	495983	14480	502254

Tabla 6.13: Resultados acertados, reales y porcentaje aciertos para Naive Bayes en el escenario 2.

ID	Cambios estado (A)	Cambios estado (R)	Porcentaje aciertos
6	146	1339	11
13	319	632	50
12	405	16339	2
10	955	1276	75
5	250	5966	4

En el escenario 2 el mejor valor de *f1-score* se logró en el lavavajillas, mientras que el mayor porcentaje de aciertos en cambios de estado se obtuvo en la jarra eléctrica. Los resultados generales de *Naive Bayes* para este escenario son bajos valores de *f1-score* y bajos porcentajes de acierto en los cambios de estado.

Resultados Escenario 3

La Tabla 6.14 presenta información acerca de los tamaños de los conjuntos de datos del escenario 3 sobre los que fue entrenado y evaluado el clasificador. A su vez, se presenta el valor de consumo *standby* escogido para cada electrodoméstico. La Tabla 6.15 muestra los valores de la matriz de confusión. Por su parte, las métricas de desempeño son expuestas en la Tabla 6.16. Se pueden observar resultados acerca de predicciones realizadas, frente a las salidas esperadas en la Tabla 6.17. En la Tabla 6.18 se puede apreciar la capacidad de detección de cambios de estado.

Tabla 6.14: Descripción de los datos para Naive Bayes en el escenario 3.

Electrodoméstico	ID	Standby	Tamaño Entrenamiento	Tamaño testeo
Lavavajillas	6	20	3641856	889273
Microondas	13	30	3583958	883110
Heladera	12	60	3584033	880709
Jarra eléctrica	10	50	3524171	870581
Lavarropas	5	50	3612179	882258

Tabla 6.15: Resultados para Naive Bayes en el escenario 3.

ID	TP	FP	TN	FN
6	15510	14052	843287	16424
13	1546	26383	854427	754
12	367332	7565	498248	7564
10	5722	27955	836115	789
5	9140	15528	840384	17206

Tabla 6.16: Métricas de desempeño para Naive Bayes en el escenario 3.

ID	F1 Score	Accuracy	Precision	Recall
6	0.5044	0.9657	0.5247	0.4857
13	0.1024	0.9693	0.0554	0.6722
12	0.9798	0.9928	0.9798	0.9798
10	0.2847	0.9700	0.1699	0.8788
5	0.3583	0.9629	0.3705	0.3469

Tabla 6.17: Resultados predichos y reales para Naive Bayes en el escenario 3.

ID	Encendidos (P)	Apagados (P)	Encendidos (R)	Apagados (R)
6	29562	859711	31934	857339
13	27929	855181	2300	880810
12	374897	505812	374896	505813
10	33677	836904	6511	864070
5	24668	857590	26346	855912

Tabla 6.18: Resultados acertados, reales y porcentaje aciertos para Naive Bayes en el escenario 3.

ID	Cambios estado (A)	Cambios estado (R)	Porcentaje aciertos
6	177	2404	7
13	329	1182	28
12	0	15128	0
10	2338	3504	67
5	200	7236	3

En este escenario el clasificador presentó altos valores en las métricas de desempeño sobre la heladera. Sin embargo, a pesar de las altas métricas obtenidas, el porcentaje de aciertos de cambios de estado para este electrodoméstico fue nulo.
El clasificador registró el mayor porcentaje de aciertos de cambios de estado en la jarra eléctrica, con un valor del 67 %.

Resultados Escenario 7

La Tabla 6.19 detalla los tamaños de los conjuntos de entrenamiento y testeo en el escenario 7, así como el valor de consumo *standby* para cada electrodoméstico. Las métricas de desempeño son presentadas en la Tabla 6.21, y en la Tabla 6.20 se muestran los valores de *TP, FP, TN, FN* que se obtuvieron para cada electrodoméstico. En la Tabla 6.22 se exponen las cantidades de predicciones y salidas reales mientras que la Tabla 6.23 ilustra la capacidad del clasificador de detectar cambios de estado.

Tabla 6.19: Descripción de los datos para Naive Bayes en el escenario 7.

Electrodoméstico	ID	Standby	Tamaño Entrenamiento	Tamaño testeo
Lavavajillas	6	20	3641856	889273
Microondas	13	30	3583598	883110
Heladera	12	60	3584033	880709
Jarra eléctrica	10	50	3524171	870581
Lavarropas	5	50	3612179	882258

Tabla 6.20: Resultados para Naive Bayes en el escenario 7.

ID	TP	FP	TN	FN
6	28066	6993	850346	3868
13	1805	8429	872381	495
12	367334	7619	498194	7562
10	4467	8194	855876	2044
5	15703	7162	848750	10643

Tabla 6.21: Métricas de desempeño para Naive Bayes en el escenario 7.

ID	F1 Score	Accuracy	Precision	Recall
6	0.8378	0.9877	0.8005	0.8788
13	0.2880	0.9898	0.1763	0.7847
12	0.9797	0.9827	0.9796	0.9798
10	0.4659	0.9882	0.3528	0.6860
5	0.6381	0.9798	0.6867	0.5960

Tabla 6.22: Resultados predichos y reales para Naive Bayes en el escenario 7.

ID	Encendidos (P)	Apagados (P)	Encendidos (R)	Apagados (R)
6	35059	854214	31934	857339
13	10234	872876	2300	880810
12	374953	505756	374896	505813
10	12661	857920	6511	864070
5	22865	859393	26346	855912

Tabla 6.23: Resultados acertados, reales y porcentaje aciertos para Naive Bayes en el escenario 7.

ID	Cambios estado (A)	Cambios estado (R)	Porcentaje aciertos
6	111	2404	5
13	337	1182	29
12	2	15128	0
10	1449	3504	41
5	143	7236	2

En este escenario el clasificador obtiene las mejores métricas de desempeño sobre la heladera y el lavavajillas. Adicionalmente, el clasificador obtuvo valores bajos para el porcentaje de aciertos en el resto de los electrodomésticos.

Desempeño general

El clasificador *Naive Bayes* constató un rendimiento malo, ya que en general obtuvo bajos valores de *f1-score* y no fue capaz de lograr altos porcentajes de acierto en cambios de estado. Las excepciones fueron la heladera y la jarra eléctrica. En el primero, el método obtuvo valores de *f1-score* cercanos a 1 en los escenarios 1,3 y 7 pero no fue capaz de predecir con exactitud sus cambios de estado. En el segundo por el contrario, las métricas fueron bajas pero el porcentaje de acierto de cambios de estado fue elevado.

6.5.2. K Nearest Neighbors

Para la implementación del clasificador *KNN* se dividió el *dataset* en conjuntos de entrenamiento y validación. Luego, el conjunto de validación se fragmentó en una proporción 50-50 para dejar un grupo de entradas reservado para la fase de testeo. En particular, se utilizó el conjunto de validación para determinar la mejor configuración de los parámetros del algoritmo (cantidad de vecinos y pesos) y los datos de testeo fueron utilizados para obtener los resultados finales. Con estos datos y teniendo en cuenta las características descritas en 6.2 se construyeron los vectores de entrada para el clasificador.

Resultados Escenario 1

En la Tabla 6.24 se muestran los mejores parámetros obtenidos para el clasificador *KNN* junto al valor de consumo *standby* para cada electrodoméstico. Los tamaños de los conjuntos de entrenamiento y testeo en el escenario 1 se detallan en la Tabla 6.25. En la Tabla 6.27 se aprecian los valores obtenidos por el clasificador en las distintas métricas de desempeño. La Tabla 6.28 indica la cantidad de predicciones y salidas reales, mientras que la Tabla 6.26 muestra los valores de *TP, FP, TN, FN* que se obtuvieron para cada electrodoméstico. La capacidad del clasificador para detectar los cambios de estados queda reflejada en la Tabla 6.29.

Tabla 6.24: Descripción de los datos y mejores parámetros para KNN en el escenario 1 para generalización a hogares vistos durante el entrenamiento.

Electrodoméstico	ID	Standby	KNN Neighbors	KNN Weights
Lavavajillas	6	20	5	Uniform
Microondas	13	30	5	Uniform
Heladera	12	60	5	Uniform
Jarra eléctrica	10	50	7	Uniform
Lavarropas	5	50	5	Uniform

Tabla 6.25: Tamaños de conjuntos de datos para KNN en el escenario 1 para generalización a hogares vistos durante el entrenamiento.

ID	Tamaño Entrenamiento	Tamaño testeo
6	2161955	270244
13	1973212	246651
12	2162831	270354
10	2124576	265572
5	2066934	258367

Tabla 6.26: Resultados para KNN en el escenario 1 para generalización a hogares vistos durante el entrenamiento.

ID	TP	FP	TN	FN
6	7589	365	262015	275
13	382	107	245987	175
12	109942	3291	152946	4175
10	486	148	264625	313
5	5944	2039	249064	1320

Tabla 6.27: Métricas de desempeño para KNN en el escenario 1 para generalización a hogares vistos durante el entrenamiento.

ID	F1 Score	Accuracy	Precision	Recall
6	0.9595	0.9976	0.9541	0.9650
13	0.7304	0.9889	0.7812	0.6858
12	0.9671	0.9724	0.9709	0.9634
10	0.6782	0.9982	0.7665	0.6082
5	0.7796	0.9869	0.7445	0.8182

Tabla 6.28: Resultados predichos y reales para KNN en el escenario 1 para generalización a hogares vistos durante el entrenamiento.

ID	Encendidos (P)	Apagados (P)	Encendidos (R)	Apagados (R)
6	7954	262290	7864	262380
13	489	246162	557	246094
12	113233	157121	114117	156237
10	634	264938	799	264773
5	7983	250384	7264	251103

Tabla 6.29: Resultados acertados, reales y porcentaje de aciertos para KNN en el escenario 1 para generalización a hogares vistos durante el entrenamiento.

ID	Cambios estado (A)	Cambios estado (R)	Porcentaje de acierto
6	445	1093	41
13	189	456	41
12	8952	16273	55
10	469	988	47
5	785	3402	23

En este escenario el clasificador presentó los mejores resultados en lo que respecta a las métricas de desempeño, sobre la heladera y el lavavajillas.
El clasificador registró el mayor porcentaje de aciertos de cambios de estado en la heladera, con un valor del 55 %. Como se puede ver en la Tabla 6.8, el clasificador obtuvo valores similares para casi todos los electrodomésticos, a excepción del lavarropas. En el escenario 1, *KNN* en general obtuvo altos valores en las métricas de desempeño como se muestra en la Tabla 6.6. En cuanto al porcentaje de acierto de cambios de estado, los valores obtenidos por el clasificador fueron similares y en el entorno del 50 %.

Resultados Escenario 2

La Tabla 6.30 ilustra los parámetros óptimos para el clasificador junto al valor de consumo *standby* discriminado por electrodoméstico. Los tamaños de los conjuntos utilizados para entrenar y evaluar el clasificador en este escenario se detallan en la Tabla 6.31. Las métricas de desempeño se visualizan en la Tabla 6.33. En la Tabla 6.34 se exhiben la cantidad de predicciones y salidas reales mientras que en la Tabla 6.32 se detalla la matriz de confusión. Los

cambios de estado predichos y acertados se exponen en la Tabla 6.35.

Tabla 6.30: Descripción de los datos y mejores parámetros para KNN en el escenario 2 para generalización a hogares vistos durante el entrenamiento.

Electrodoméstico	ID	Standby	KNN Neighbors	KNN Weights
Lavavajillas	6	20	7	Distance
Microondas	13	30	5	Uniform
Heladera	12	60	7	Uniform
Jarra eléctrica	10	50	5	Uniform
Lavarropas	5	50	3	Uniform

Tabla 6.31: Tamaños de conjuntos de datos para KNN en el escenario 2 para generalización a hogares vistos durante el entrenamiento.

ID	Tamaño Entrenamiento	Tamaño testeo
6	2161955	270244
13	1973212	246651
12	2162831	270354
10	2124576	265572
5	2066934	258367

Tabla 6.32: Resultados para KNN en el escenario 2 para generalización a hogares vistos durante el entrenamiento.

ID	TP	FP	TN	FN
6	1300	832	261548	6564
13	138	235	245859	419
12	70215	52562	103675	43902
10	152	281	264492	647
5	1296	4825	246278	5968

Tabla 6.33: Métricas de desempeño para KNN en el escenario 2 para generalización a hogares vistos durante el entrenamiento.

ID	F1 Score	Accuracy	Precision	Recall
6	0.2601	0.9726	0.6097	0.1653
13	0.2967	0.9973	0.3699	0.2477
12	0.5927	0.6431	0.5718	0.6152
10	0.2467	0.9965	0.3510	0.1902
5	0.1936	0.9582	0.2117	0.1784

Tabla 6.34: Resultados predichos y reales para KNN en el escenario 2 para generalización a hogares vistos durante el entrenamiento.

ID	Encendidos (P)	Apagados (P)	Encendidos (R)	Apagados (R)
6	2132	268112	7864	262380
13	373	246278	557	246094
12	122777	147577	114117	156237
10	433	265139	799	264773
5	6121	252246	7264	251103

Tabla 6.35: Resultados acertados, reales y porcentaje de aciertos para KNN en el escenario 2 para generalización a hogares vistos durante el entrenamiento.

ID	Cambios estado (A)	Cambios estado (R)	Porcentaje de acierto
6	67	1093	6
13	86	456	19
12	6098	16273	37
10	195	988	20
5	293	3402	9

En este escenario el clasificador presentó los mejores resultados en lo que respecta a las métricas de desempeño sobre la heladera. En general, el clasificador presentó valores bajos en las métricas de desempeño para todos los electrodomésticos.
El clasificador obtuvo un bajo desempeño en la detección de cambios de estado, donde el electrodoméstico sobre el cual se destacó fue la heladera con un porcentaje de detección del 37 %.

Resultados Escenario 3

Los parámetros que permitieron alcanzar los mejores resultados, junto al valor de consumo *standby* para cada electrodoméstico son expuestos en la Tabla 6.36. La Tabla 6.37 muestra las dimensiones de los conjuntos de entrenamiento y testeo utilizados por el clasificador en este escenario. En la Tabla 6.38 se pueden observar los valores de la matriz de confusión obtenidos para cada electrodoméstico, estos valores determinan las métricas de desempeño ilustradas en la Tabla 6.39. Por su parte, la Tabla 6.40 presenta la comparativa entre predicciones realizadas por el clasificador y las salidas reales. Finalmente, es en

la Tabla 6.41 que se presentan los porcentajes de acierto de cambios de estado para cada electrodoméstico.

Tabla 6.36: Descripción de los datos y mejores parámetros para KNN en el escenario 3 para generalización a hogares vistos durante el entrenamiento.

Electrodoméstico	ID	Standby	KNN Neighbors	KNN Weights
Lavavajillas	6	20	7	uniform
Microondas	13	30	7	uniform
Heladera	12	60	5	uniform
Jarra eléctrica	10	50	7	uniform
Lavarropas	5	50	7	uniform

Tabla 6.37: Tamaños de conjuntos de datos para KNN en el escenario 3 para generalización a hogares vistos durante el entrenamiento.

ID	Tamaño Entrenamiento	Tamaño testeo
6	3641856	444636
13	3583958	441555
12	3584033	440354
10	3524171	435290
5	3612179	441129

Tabla 6.38: Resultados para KNN en el escenario 3 para generalización a hogares vistos durante el entrenamiento.

ID	TP	FP	TN	FN
6	15521	643	428040	432
13	858	112	440286	299
12	172235	4384	260191	3545
10	2384	418	431609	879
5	11463	2290	425649	1727

Tabla 6.39: Métricas de desempeño para KNN en el escenario 3 para generalización a hogares vistos durante el entrenamiento.

ID	F1 Score	Accuracy	Precision	Recall
6	0.9665	0.9976	0.9602	0.9729
13	0.8067	0.9990	0.8845	0.7415
12	0.9774	0.9819	0.9751	0.9798
10	0.7861	0.9970	0.8508	0.7306
5	0.8510	0.9909	0.8335	0.8691

Tabla 6.40: Resultados predichos y reales para KNN en el escenario 3 para generalización a hogares vistos durante el entrenamiento.

ID	Encendidos (P)	Apagados (P)	Encendidos (R)	Apagados (R)
6	16164	428472	15953	428683
13	970	440585	1157	440398
12	176619	263736	175780	264575
10	2802	432488	3263	432027
5	13753	427376	13190	427939

Tabla 6.41: Resultados acertados, reales y porcentaje de aciertos para KNN en el escenario 3 para generalización a hogares vistos durante el entrenamiento.

ID	Cambios estado (A)	Cambios estado (R)	Porcentaje de acierto
6	257	1340	19
13	180	738	24
12	1102	7358	15
10	712	1852	38
5	379	3472	11

En este escenario el clasificador presentó los mejores resultados en lo que respecta a las métricas de desempeño sobre la heladera y el lavavajillas.
En contraposición con los altos valores obtenidos por el clasificador para las métricas de desempeño, el clasificador arrojó malos resultados en lo que respecta al porcentaje de acierto de cambios de estado.

Resultados Escenario 7

En la Tabla 6.42 se muestran los parámetros que hicieron al clasificador *KNN* más preciso junto al valor de consumo *standby* para cada electrodoméstico. En la Tabla 6.21 se muestran las dimensiones de los conjuntos de entrenamiento y testeo que se emplearon en este escenario. Se presentan en las Tablas 6.45 y 6.44 las métricas de desempeño y matriz de confusión respectivamente. La Tabla 6.46 ilustra la cantidad de predicciones y salidas reales. El rendimiento del clasificador para detectar cambios de estado puede verse en la Tabla 6.47.

Tabla 6.42: Descripción de los datos y mejores parámetros para KNN en el escenario 7.

Electrodoméstico	ID	Standby	KNN Neighbors	KNN Weights
Lavavajillas	6	20	3	uniform
Microondas	13	30	3	uniform
Heladera	12	60	5	uniform
Jarra eléctrica	10	50	7	uniform
Lavarropas	5	50	3	uniform

Tabla 6.43: Tamaños de conjuntos de datos para KNN en el escenario 7.

ID	Tamaño Entrenamiento	Tamaño testeo
6	3641856	444637
13	3583598	441555
12	3583674	440355
10	3524171	435291
5	3612179	441129

Tabla 6.44: Resultados para KNN en el escenario 7.

ID	TP	FP	TN	FN
6	17733	591	425877	436
13	864	401	439816	474
12	173082	3029	261546	2698
10	2905	472	431281	633
5	18324	1894	426547	1328

Tabla 6.45: Métricas de desempeño para KNN en el escenario 7.

ID	F1 Score	Accuracy	Precision	Recall
6	0.97	0.99	0.97	0.98
13	0.67	0.99	0.68	0.65
12	0.98	0.99	0.98	0.98
10	0.84	0.99	0.86	0.82
5	0.88	0.99	0.86	0.90

Tabla 6.46: Resultados predichos y reales para KNN en el escenario 7.

ID	Encendidos (P)	Apagados (P)	Encendidos (R)	Apagados (R)
6	18324	426313	18169	426468
13	1265	440290	1338	440217
12	176111	264244	175780	264575
10	3377	431914	3538	431753
5	13254	427875	12688	428441

Tabla 6.47: Resultados acertados, reales y porcentaje de aciertos para KNN en el escenario 7.

ID	Cambios estado (A)	Cambios estado (R)	Porcentaje de acierto
6	411	1340	31
13	159	738	22
12	3636	7358	49
10	984	1852	53
5	722	3472	21

En el escenario 7 el clasificador presentó los valores más altos en las métricas de desempeño sobre la heladera y el lavavajillas. Para el resto de los electrodomésticos, si bien los valores de las métricas de desempeño están por debajo de la heladera y el lavavajillas, igualmente fueron altos.
En lo que refiere al porcentaje de aciertos de cambios de estado, el clasificador obtuvo el mayor valor en la jarra eléctrica con un 53 %.

Desempeño general

El clasificador *KNN* presentó en general valores altos en las métricas de desempeño en todos los electrodomésticos y todos los escenarios, con la a excepción de los resultados obtenidos en el escenario 2. En cuanto a la capacidad de detectar cambios de estados, el clasificador tuvo en general un pobre rendimiento.

6.5.3. Perceptrón multicapa

El perceptrón multicapa fue evaluado sobre el escenario 4 y requirió del ajuste de la cantidad de neuronas presentes en la capa oculta (*hidden units*), el tamaño del lote de entrenamiento *(batchsize)*, la cantidad de iteraciones por

lote *(iterations)* y la taza de aprendizaje *(learning rate)*.

En las tablas 6.48 y 6.49 se muestran los valores de consumo *standby* empleados y las dimensiones de los conjuntos de datos respectivamente. Las tablas 6.50, 6.51, 6.52, 6.53, 6.54, y 6.55 detallan los resultados obtenidos por la red en el escenario 4 con la siguiente configuración paramétrica: *Hidden units = 5, Batchsize = 2000, Iterations = 100, Learning rate = 0.005.*

Tabla 6.48: Descripción de los datos para MLP en el escenario 4.

Electrodoméstico	ID	Standby
Lavavajillas	6	20
Microondas	13	30
Heladera	12	60
Jarra eléctrica	10	50
Lavarropas	5	50

Tabla 6.49: Tamaños de conjuntos de datos para MLP en el escenario 4.

ID	Tamaño Entrenamiento	Tamaño testeo
6	3641856	444637
13	3583598	441555
12	3583674	440355
10	3524171	435291
5	3612179	441129

Tabla 6.50: Resultados para MLP en el escenario 4.

ID	TP	FP	TN	FN
6	17499	670	425798	670
13	969	369	439848	369
12	172101	3679	260896	3679
10	0	0	431753	3538
5	10952	1736	426705	1736

Tabla 6.51: Métricas de desempeño para MLP en el escenario 4.

ID	F1 Score	Accuracy	Precision	Recall
6	0.9631	0.9970	0.9631	0.9631
13	0.7242	0.9983	0.7242	0.7242
12	0.9791	0.9833	0.9791	0.9791
10	0	0.9919	0	0
5	0.8632	0.9921	0.8632	0.8632

Tabla 6.52: Resultados predichos y reales para MLP en el escenario 4.

ID	Encendidos (P)	Apagados (P)	Encendidos (R)	Apagados (R)
6	18169	426468	18169	426468
13	1338	440217	1338	440217
12	175780	264575	175780	264575
10	0	435291	3538	431753
5	12688	428441	12688	428441

Tabla 6.53: Resultados predichos, acertados y reales para MLP en el escenario 4.

ID	Cambios estado (P)	Cambios estado (A)	Cambios estado (R)
6	2404	0	2404
13	0	0	1182
12	15128	0	15128
10	3066	1155	3504
5	7234	4	7236

Tabla 6.54: Resultados de cambios de estado (C-E) reales y acertados para MLP en el escenario 4.

ID	C-E encendido (R)	C-E encendido (A)	C-E apagado (R)	C-E apagado (A)
6	670	0	670	0
13	369	0	369	0
12	3679	0	3679	0
10	926	0	926	0
5	1736	0	1736	0

Tabla 6.55: Porcentaje cambios de estado acertados para MLP en el escenario 4.

ID	% Acierto
6	0
13	0
12	0
10	0
5	0

Desempeño general

A pesar de que se evaluaron diferentes configuraciones para los parámetros *hidden units*, *(batchsize)*, *(iterations)* y *(learning rate)*, siempre se obtuvo el mismo resultado: *overfitting* sobre la variable *ENCENDIDO_ANTERIOR*. La red aprende a retornar el valor indicado por dicha característica desarrollando el siguiente comportamiento: la salida de la red será correcta en la mayoría de sus predicciones y será errónea en los momentos que ocurre un cambio de estado, detectándolo en el instante posterior.
El *overfitting* se observó en todos los escenarios que utilizan como característica de entrada el estado anterior del electrodoméstico, o los tiempos de encendido y apagado. Estos últimos provocan el mismo resultado ya que sus valores se hacen cero al momento de producirse el cambio de estado. En la gráfica 6.1 se ilustra el problema, muestra los valores reales y los predichos por el clasificador para la heladera en los últimos cien vectores de testeo.

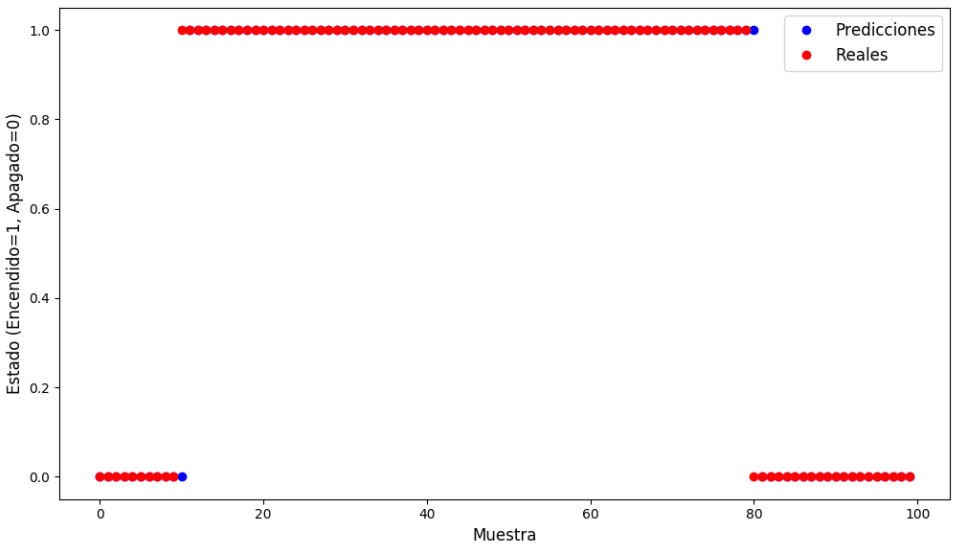


Figura 6.1: Últimas cien predicciones de MLP sobre heladera en el escenario 4.

La red se testeó sobre los escenarios que no incluían las variables responsables de *overfitting* en sus vectores de entrada, pero los resultados no fueron alentadores y por lo tanto se profundizó en el uso de redes con memoria.

6.5.4. Redes Neuronales LSTM

Las redes LSTM se evaluaron en los escenarios 4, 7, 8 y 9 de la variante binaria del problema, y en los escenarios 5 y 6 de la variante con múltiples niveles de consumo. A continuación se presenta la configuración paramétrica de la red y los resultados obtenidos por este clasificador.

Arquitectura de la red

Las arquitecturas expuestas en la tesis de Kelly [7], se utilizaron como base para comenzar a experimentar con los distintos parámetros que son parte de la configuración de una red neuronal. Se combinaron diversos tipos de capas, cantidad de neuronas, funciones de costo, optimizadores, etc. Se evaluaron los resultados obtenidos por cada una de las arquitecturas de redes neuronales expuestas en la sección 5.2.4, para finalmente optar por una arquitectura con la siguiente configuración:

1. LSTM bidireccional, 128 neuronas y función de activación *Tanh*
2. LSTM bidireccional, 256 neuronas y función de activación *Tanh*
3. LSTM simple, 128 neuronas y función de activación *Tanh*
4. Completamente conectada, 1 neurona y función de activación *Sigmoid*

A su vez, dentro de los escenarios vinculados con la variante binaria se utilizaron entropía cruzada binaria como función de costo, *Adam* como algoritmo de optimización y se estableció en 15 épocas la cantidad de iteraciones de entrenamiento. Para la variante de múltiples niveles de consumo se ratificó el algoritmo de optimización, pero se aumentó a 25 las iteraciones de entrenamiento y se pasó a utilizar entropía cruzada categórica dispersa (*sparse categorical cross entropy*) como función de costo.

Resultados Escenario 4

En la Tabla 6.56 se muestran los valores de consumo *standby* y los distintos tamaños de ventana de memoria (*timesteps*) utilizados. La Tabla 6.57 presenta

las dimensiones de los conjuntos de entrenamiento, validación y testeo. La Tabla 6.58 detalla los valores de la matriz de confusión, mientras que en la Tabla 6.59 se pueden observar las métricas de desempeño asociadas. Las tablas 6.60, 6.61 y 6.62 contabilizan los encendidos, apagados y cambios de estado, tanto acertados como reales.

Tabla 6.56: Descripción de los datos y mejores parámetros para LSTM en el escenario 4 para generalización a hogares vistos durante el entrenamiento.

Electrodoméstico	ID	Standby	Timesteps
Lavavajillas	6	20	5
Microondas	13	30	5
Heladera	12	60	3
Jarra eléctrica	10	50	3
Lavarropas	5	50	3

Tabla 6.57: Tamaños de conjuntos de datos para LSTM en el escenario 4 para generalización a hogares vistos durante el entrenamiento.

ID	Tamaño entrenamiento	Tamaño validación	Tamaño testeo
6	3641851	444631	444631
13	3583591	441541	441541
12	3583672	440350	440353
10	3524167	435286	435289
5	3612175	441127	441127

Tabla 6.58: Resultados para LSTM en el escenario 4 para generalización a hogares vistos durante el entrenamiento.

ID	TP	FP	TN	FN
6	17681	447	426015	488
13	956	79	440134	382
12	174080	2242	262331	1700
10	3092	169	431582	446
5	11702	1620	426819	986

Tabla 6.59: Métricas de desempeño para LSTM en el escenario 4 para generalización a hogares vistos durante el entrenamiento.

ID	F1 Score	Accuracy	Precision	Recall
6	0.97	0.99	0.97	0.97
13	0.81	0.99	0.92	0.72
12	0.99	0.99	0.99	0.99
10	0.91	0.99	0.95	0.87
5	0.90	0.99	0.88	0.92

Tabla 6.60: Resultados predichos y reales para LSTM en el escenario 4 para generalización a hogares vistos durante el entrenamiento.

ID	Encendidos (P)	Apagados (P)	Encendidos (R)	Apagados (R)
6	18128	426503	18169	426462
13	1035	440516	1338	440213
12	176322	264031	175780	264573
10	3261	432028	3538	431751
5	13322	4278045	12688	428439

Tabla 6.61: Resultados de cambios de estado (C-E) para LSTM en el escenario 4 para generalización a hogares vistos durante el entrenamiento.

ID	C-E encendido (R)	C-E encendido (A)	C-E apagado (R)	C-E apagado (A)
6	670	263	670	207
13	369	31	369	177
12	3679	2298	3679	2366
10	926	542	926	749
5	1736	500	1736	352

Tabla 6.62: Porcentaje de cambios de estado (C-E) acertados para LSTM en el escenario 4 para generalización a hogares vistos durante el entrenamiento.

ID	Porcentaje de acierto
6	35.0
13	28.0
12	63.0
10	69.0
5	24.0

La red LSTM en el escenario 4 obtuvo altos valores de *f1-score* para todos los electrodomésticos. Fue capaz de detectar más de la mitad de los cambios de estado que presentaron la heladera y la jarra eléctrica, con porcentajes de acierto de 63 % y 69 % respectivamente.

Resultados Escenario 7

La Tabla 6.63 ilustra la cantidad de *timesteps* óptima para cada electrodoméstico junto al valor de consumo *standby*. Los tamaños de los conjuntos utilizados para entrenar, evaluar y testear la red LSTM se encuentran reflejados en la Tabla 6.64. Las Tablas 6.65 y 6.66 muestran la matriz de confusión y las métricas de desempeño respectivamente. En la Tabla 6.67 se exhiben la cantidad de predicciones y salidas reales. En las Tabla 6.68 se ilustra la cantidad de cambios de estados, tanto acertados como reales, mientras que en la Tabla 6.69 se indica el porcentaje de aciertos de los mismos.

Tabla 6.63: Descripción de los datos y mejores parámetros para LSTM en el escenario 7 para generalización a hogares vistos durante el entrenamiento.

Electrodoméstico	ID	Standby	Timesteps
Lavavajillas	6	20	3
Microondas	13	30	5
Heladera	12	60	3
Jarra eléctrica	10	50	3
Lavarropas	5	50	10

Tabla 6.64: Tamaños de conjuntos de datos para LSTM en el escenario 7 para generalización a hogares vistos durante el entrenamiento.

ID	Tamaño entrenamiento	Tamaño validación	Tamaño testeo
6	3641854	444634	444634
13	3583581	441541	441541
12	3583672	440350	440353
10	3524167	435286	435289
5	3612161	441111	441111

Tabla 6.65: Resultados para LSTM en el escenario 7 para generalización a hogares vistos durante el entrenamiento.

ID	TP	FP	TN	FN
6	17565	246	426219	604
13	1026	257	439956	312
12	173738	2185	262388	2042
10	3179	206	431545	359
5	11774	1305	427118	914

Tabla 6.66: Métricas de desempeño para LSTM en el escenario 7 para generalización a hogares vistos durante el entrenamiento.

ID	F1 Score	Accuracy	Precision	Recall
6	0.98	0.99	0.99	0.97
13	0.78	0.99	0.80	0.77
12	0.99	0.99	0.99	0.99
10	0.92	0.99	0.94	0.90
5	0.91	0.99	0.90	0.93

Tabla 6.67: Resultados predichos y reales para LSTM en el escenario 7 para generalización a hogares vistos durante el entrenamiento.

ID	Encendidos (P)	Apagados (P)	Encendidos (R)	Apagados (R)
6	17811	426823	18169	426465
13	1284	440267	1338	440213
12	175923	264430	175780	264573
10	3385	431904	3538	431751
5	13079	428032	12688	428423

Tabla 6.68: Resultados de cambios de estado (C-E) para LSTM en el escenario 7 para generalización a hogares vistos durante el entrenamiento.

ID	C-E encendido (R)	C-E encendido (A)	C-E apagado (R)	C-E apagado (A)
6	670	272	670	205
13	369	97	369	202
12	3679	2657	3679	2579
10	926	629	926	775
5	1736	507	1736	590

Tabla 6.69: Porcentaje de cambios de estado (C-E) acertados para LSTM en el escenario 7 para generalización a hogares vistos durante el entrenamiento.

ID	Porcentaje de acierto
6	35.0
13	40.0
12	71.0
10	75.0
5	31.0

La red LSTM obtuvo en las métricas de desempeño valores en su mayoría por encima del 90 %, logrando el mejor desempeño en la heladera con un *f1-score* de 0.99 %. En lo que respecta a la detección de cambios de estados, la red obtuvo sus mejores desempeños sobre la jarra eléctrica y la heladera, con valores de 75 % y 71 % respectivamente.

Resultados Escenario 8

Este escenario es una variante del escenario 7 del cual se removieron las características *ENCENDIDO_ANTERIOR*, *TIEMPO_ENCENDIDO* y *TIEMPO_APAGADO* pensando que podían ser motivo de *overfitting*. Se constató que el *overfitting* era un problema únicamente en la red de perceptrones multicapa sin memoria, donde se observó que el aprendizaje se orienta a retornar el valor de la característica *ENCENDIDO_ANTERIOR*. Por otra parte, las redes LSTM no presentaron este problema, de hecho empeoraron su desempeño al remover las características antes mencionadas y por lo tanto no se profundizó en la evaluación experimental de este escenario.

Resultados Escenario 9

Con el objetivo de aumentar la detección de cambios de estado por parte del clasificador, se entrenaron los algoritmos con subconjuntos de datos de entrada correspondientes a la semana y el mes donde se produjeron la mayor cantidad de cambios de estados. Los resultados preliminares que se obtuvieron fueron inferiores a los obtenidos en el escenario 7 y por tanto no se prosiguió en la evaluación experimental de este escenario.

Resultados Escenario 5

Fueron solo dos los electrodomésticos escogidos para la realización de las evaluaciones preliminares del desempeño de este clasificador dentro de este escenario. Esta selección estuvo basada en que la heladera y el lavarropas arrojaban a nivel general los mejores y peores resultados respectivamente para los clasificadores utilizados en la variante binaria del problema.

La Tabla 6.70 expone los consumos *standby* y los tamaños de venta escogidos para cada electrodoméstico. Por su parte, la Tabla 6.71 muestra los tamaños de los conjuntos utilizados para el entrenamiento y evaluación de este clasificador dentro de este escenario. Las tablas 6.72 y 6.73 ilustran los distintos niveles de consumo presentes en cada electrodoméstico. Los métricas de desempeño obtenidas para cada uno de los electrodomésticos son presentadas en las tablas 6.74 y 6.75. En lo que a cambios de estado respecta, información sobre el desempeño de cada electrodoméstico en este aspecto es indicada en las tablas 6.76 y 6.77.

Tabla 6.70: Descripción de los datos y mejores parámetros para LSTM en el escenario 5 para generalización a hogares vistos durante el entrenamiento.

Electrodoméstico	ID	Standby	Timesteps
Heladera	12	60	5
Lavarropas	5	50	5

Tabla 6.71: Tamaños de conjuntos de datos para LSTM en el escenario 5 para generalización a hogares vistos durante el entrenamiento.

ID	Tamaño Entrenamiento	Tamaño Validación	Tamaño testeo
12	3583669	440347	440347
5	3612169	441121	441121

Tabla 6.72: Estados definidos de la heladera para LSTM en el escenario 5 para generalización a hogares vistos durante el entrenamiento.

Clase	Consumo Inicial	Consumo Final
Estado 0	0	62
Estado 1	63	223
Estado 2	224	3272
Estado 3	3273	3323

Tabla 6.73: Estados definidos del lavarropas para LSTM en el escenario 5 para generalización a hogares vistos durante el entrenamiento.

Clase	Consumo Inicial	Consumo Final
Estado 0	0	141
Estado 1	142	1740
Estado 2	1741	1949
Estado 3	1950	2171
Estado 4	2172	3948
Estado 5	3949	3999

Tabla 6.74: Métricas de desempeño de la heladera para LSTM en el escenario 5 para generalización a hogares vistos durante el entrenamiento.

Clase	F1 Score	Precision	Recall
Estado 0	0.98	0.98	0.98
Estado 1	0.97	0.96	0.97
Estado 2	0.91	0.95	0.87
Estado 3	-	-	-

Tabla 6.75: Métricas de desempeño del lavarropas para LSTM en el escenario 5 para generalización a hogares vistos durante el entrenamiento.

Clase	F1 Score	Precision	Recall
Estado 0	0.99	0.99	0.99
Estado 1	0.78	0.87	0.71
Estado 2	0.0	0.0	0.0
Estado 3	0.89	0.90	0.87
Estado 4	0.54	0.61	0.47
Estado 5	0.0	0.0	0.0

Tabla 6.76: Resultados de cambios de estado (C-E) en la heladera para LSTM en el escenario 5 para generalización a hogares vistos durante el entrenamiento.

Clase	C-E Reales	C-E Predichos	C-E acertados	Porcentaje de Acierto
Estado 0	3679	5293	513	14.0
Estado 1	3646	5117	638	18.0
Estado 2	442	318	25	6.0
Estado 3	0	0	0	0.0

Tabla 6.77: Resultados de cambios de estado (C-E) en el lavarropas para LSTM en en el escenario 5 para generalización a hogares vistos durante el entrenamiento.

Clase	C-E Reales	C-E Predichos	C-E Acertados	Porcentaje de acierto
Estado 0	486	443	131	27.0
Estado 1	535	262	2	0.37
Estado 2	22	53	0	0.0
Estado 3	301	284	24	8.0
Estado 4	220	124	11	5.0
Estado 5	0	1	0	0.0

Este escenario particular fue planteado para intentar resolver la variante del problema de múltiples niveles de consumo. Esta línea de trabajo se continuó hasta que se realizaron los primeros ensayos y sus respectivas evaluaciones experimentales. Allí se constató que los resultados obtenidos no eran los esperados.

Resultados Escenario 6

Al igual que sucedió en el escenario 5, los resultados iniciales fueron negativos y se corroboró que esta línea de trabajo no es viable, por lo tanto no se profundizó en la evaluación experimental del escenario 6.

Desempeño general

En la variante binaria del problema las redes LSTM lograron elevados valores en cuanto a las métricas de *accuracy, precision, recall y f1-score*. La capacidad de detección de estados fue buena para la heladera y la jarra eléctrica en todos los escenarios de evaluación. En el resto de los electrodomésticos no se

lograron resultados elevados en el porcentaje de acierto de cambios de estado. Por otro lado, para la variante de múltiples niveles de consumo, la red tuvo en general malos resultados, siendo particularmente difícil la tarea de detección de cambios de estado.

6.6. Resultados sobre hogares no vistos durante el entrenamiento

Como se mencionó en la introducción del capítulo, en este apartado se presentan los resultados obtenidos por los clasificadores al enfrentarse a conductas de consumo eléctrico no conocidas, es decir no proporcionadas en las etapas de entrenamiento.

6.6.1. Introducción

Las conductas de consumo eléctrico dentro de los hogares depende de factores como la composición del núcleo familiar, el nivel socio económico, el clima de la ciudad a la que pertenece, u otros aspectos culturales, entre los más relevantes. Lo que se busca estudiar en esta subsección es el desempeño de los algoritmos de aprendizaje automático ante patrones de uso desconocidos. Más precisamente, se entrenan todos los clasificadores con datos de un hogar específico, para luego medir la calidad de las predicciones con información de otro. Tal como se indicó en la sección 6.1.1, los datos de entrenamiento provienen del hogar uno y al momento de evaluar se utilizaron los datos del hogar dos (validación y testeo). Los escenarios sobre los cuales se realizaron los experimentos fueron el escenario 3, escenario 4 y el escenario 7, haciendo uso de todos los clasificadores con excepción de MLP debido a los malos resultados obtenidos por este algoritmo, reportados en la sección 6.5. Los electrodomésticos seleccionados para las evaluaciones fueron la heladera y el microondas, los cuales en la evaluación con hogares vistos durante entrenamiento obtuvieron los mejores y peores resultados respectivamente.

6.6.2. Naive Bayes

Naive Bayes fue evaluado en los escenarios 3 y 7 y los resultados obtenidos se exponen a continuación.

Resultados Escenario 3

La Tabla 6.78 presenta las métricas de desempeño obtenidas por este clasificador dentro de este escenario. Mientras que en la Tabla 6.79 se expone la capacidad del clasificador para acertar cambios de estado.

Tabla 6.78: Métricas de desempeño para Naive Bayes en el escenario 3 para generalización a hogares no conocidos.

Electrodoméstico	ID	F1 Score	Accuracy	Precision	Recall
Heladera	14	0.9607	0.9727	0.9607	0.9607
Microondas	15	0.0795	0.9712	0.0447	0.3539

Tabla 6.79: Resultados sobre cambios de estado para Naive Bayes en el escenario 3 para generalización a hogares no conocidos.

ID	Cambios estado (A)	Cambios estado (R)	Porcentaje de acierto
14	0	1887	0
15	25	118	21

El clasificador presentó en el escenario 3 resultados altos en las métricas de desempeño sobre la heladera. Sin embargo, en lo que refiere al microondas, los resultados fueron bajos. En la detección de cambios de estado, el clasificador *Naive Bayes* obtuvo 0 % sobre la heladera y 21 % sobre el microondas.

Resultados Escenario 7

En la Tabla 6.80 se presentan las métricas de desempeño obtenidas por el clasificador *Naive Bayes* en este escenario. La Tabla 6.81 ilustra los cambios de estado acertados, reales y el porcentaje de aciertos del clasificador.

Tabla 6.80: Métricas de desempeño para Naive Bayes en el escenario 7 para generalización a hogares no conocidos.

Electrodoméstico	ID	F1 Score	Accuracy	Precision	Recall
Heladera	14	0.9601	0.9722	0.9594	0.9608
Microondas	15	0.3438	0.9906	0.2283	0.6954

Tabla 6.81: Resultados sobre cambios de estado para Naive Bayes en el escenario 7 para generalización a hogares no conocidos.

ID	Cambios estado (A)	Cambios estado (R)	Porcentaje de acierto
14	0	1887	0
15	48	118	41

El clasificador obtuvo bajas tasas de acierto de cambios de estado, siendo nula para la heladera y con un valor de 41 % para el microondas. En cuanto a las métricas de desempeño, el clasificador presentó altos valores para la heladera, pero bajos para el microondas.

Desempeño general

El clasificador mostró un comportamiento similar en ambos escenarios. Las métricas de desempeño obtenidas por el clasificador fueron altas para la heladera pero bajas para el microondas, a excepción de la *accuracy*. En cuanto a la capacidad de detectar los cambios de estado, el clasificador mostró un mal rendimiento, obteniendo valor nulo para ambos escenarios sobre la heladera y muy bajos valores sobre el microondas.

6.6.3. K Nearest Neighbors

El clasificador *KNN* fue evaluado sobre los escenarios 3 y 7 y los resultados obtenidos se presentan a continuación.

Resultados Escenario 3

La Tabla 6.82 ilustra los parámetros óptimos para el clasificador *KNN*. En la Tabla 6.83 se pueden apreciar los valores de las métricas de desempeño obtenidas por el clasificador. La Tabla 6.84 exhibe el desempeño del clasificador en cuanto a la capacidad de detección de cambios de estado.

Tabla 6.82: Parámetros óptimos para KNN en el escenario 3 para generalización a hogares no conocidos.

Electrodoméstico	ID	KNN Neighbors	KNN Weights
Heladera	14	5	Uniform
Microondas	15	7	Uniform

Tabla 6.83: Métricas de desempeño para KNN en el escenario 3 para generalización a hogares no conocidos.

ID	F1 Score	Accuracy	Precision	Recall
14	0.9263	0.9516	0.9838	0.8752
15	0.8384	0.9989	0.8888	0.7933

Tabla 6.84: Resultados sobre cambios de estado para KNN en el escenario 3 para generalización a hogares no conocidos.

ID	Cambios estado (A)	Cambios estado (R)	Porcentaje de acierto
14	139	952	15
15	20	52	38

En el Escenario 3 el clasificador presentó muy buenos resultados en las métricas de desempeño tanto para la heladera como para el microondas. En lo que refiere a la detección de cambios de estados el clasificador presentó malos resultados.

Resultados Escenario 7

La Tabla 6.85 muestra los parámetros óptimos para el clasificador *KNN* para los electrodomésticos que se evalúan en este experimento. En la Tabla 6.86 se muestran los valores obtenidos por el clasificador en cuanto a las métricas de desempeño. La Tabla 6.87 muestra el desempeño del clasificador en su capacidad de detectar los cambios de estado.

Tabla 6.85: Parámetros óptimos para KNN en Escenario 7 para generalización a hogares no conocidos.

Electrodoméstico	ID	KNN Neighbors	KNN Weights
Heladera	14	5	Uniform
Microondas	15	3	Uniform

Tabla 6.86: Métricas de desempeño para KNN en Escenario 7 para generalización a hogares no conocidos.

ID	F1 Score	Accuracy	Precision	Recall
14	0.9630	0.9742	0.9660	0.9600
15	0.7300	0.9984	0.7448	0.7156

Tabla 6.87: Resultados sobre cambios de estado para KNN en escenario 7 para generalización a hogares no conocidos.

ID	Cambios estado (A)	Cambios estado (R)	Porcentaje de acierto
14	181	952	19
15	17	52	32

En el escenario 7 el clasificador presentó valores altos en las métricas de desempeño sobre la heladera. Por el contrario, en el microondas no fueron buenos. En cuanto a la detección de cambios de estado, el clasificador presentó malos resultados para ambos electrodomésticos.

Desempeño general

El clasificador *KNN* logró altos valores en las métricas de desempeño para ambos electrodomésticos y en ambos escenarios. Sin embargo, el porcentaje de acierto de cambios de estado fue bajo en todos los casos.

6.6.4. Redes LSTM

Las redes neuronales LSTM fueron evaluadas sobre los escenarios 4 y 7, los resultados obtenidos se pueden observar a continuación.

Resultados Escenario 4

La Tabla 6.88 ilustra las métricas de desempeño obtenidas por las redes LSTM para el escenario 4. En la Tabla 6.89 se muestran los porcentajes de aciertos en cambios de estado para ca uno de los electrodomésticos.

Tabla 6.88: Métricas de desempeño para LSTM en el escenario 3 para generalización a hogares no conocidos.

Electrodoméstico	ID	F1 Score	Accuracy	Precision	Recall
Heladera	14	0.9856	0.9899	0.9837	0.9876
Microondas	15	0.8118	0.9989	0.8200	0.8039

Tabla 6.89: Resultados sobre cambios de estado para LSTM en el escenario 3 para generalización a hogares no conocidos.

ID	Cambios estado (A)	Cambios estado (R)	Porcentaje de acierto
14	735	952	77
15	28	52	54

Las redes LSTM presentaron resultados altos tanto en las métricas de desempeño como en la detección de cambios de estado para ambos electrodomésticos.

Resultados Escenario 7

La Tabla 6.90 presenta las métricas de desempeño obtenidas por las redes LSTM. En la Tabla 6.91 se pueden ver los valores asociados a la capacidad de detección de cambios de estado.

Tabla 6.90: Métricas de desempeño para LSTM en el escenario 7 para generalización a hogares no conocidos.

Electrodoméstico	ID	F1 Score	Accuracy	Precision	Recall
Heladera	14	0.9799	0.9859	0.9784	0.9813
Microondas	15	0.8629	0.9992	0.8947	0.8333

Tabla 6.91: Resultados sobre cambios de estado para LSTM en el escenario 7 para generalización a hogares no conocidos.

ID	Cambios estado (A)	Cambios estado (R)	Porcentaje de acierto
14	637	952	67
15	27	52	52

En el escenario 7 las redes LSTM lograron altos valores en las métricas de desempeño y los porcentajes de aciertos en cambios de estado.

Desempeño general

En lo que respecta a los resultados obtenidos en este clasificador, se puede decir que fueron altamente satisfactorios. Por un lado, las métricas de desempeño tuvieron valores muy altos, mientras que los porcentajes de aciertos en cambios de estado fueron muy buenos.

6.7. Análisis comparativo de los métodos implementados

Como los clasificadores desarrollados son específicos para cada electrodoméstico y cada uno se ejecuta sobre diferentes escenarios, con el fin de contrastar los resultados se elaboró un ranking agregado por electrodoméstico. En este ranking se exponen los resultados de cada clasificador para cada uno de los escenarios presentados en la Sección 6.2 sobre los que ejecutó. Las dos métricas que se utilizaron para realizar la evaluación fueron el *f1-score* y el porcentaje de aciertos en cambios de estado. La métrica *f1-score* es una medida ponderada de la *precision* y el *recall* que aporta información sobre ambas métricas. Por otra parte, el instante en el cual un electrodoméstico cambia de estado es la base del problema de desagregación energética. Por lo tanto, es necesario considerar este factor al momento de comparar los clasificadores implementados.

La Tabla 6.92 ilustra un ranking elaborado en base a los resultados obtenidos por los clasificadores en base a la métrica *f1-score* y porcentaje de aciertos en cambios de estado sobre el lavavajillas.

Tabla 6.92: Ranking de resultados obtenidos por los clasificadores en el lavavajillas.

Posición	Clasificador	Escenario	F1-Score	Porcentaje acierto cambios de estado
1	KNN	1	0.96	41.0
2	LSTM	7	0.98	35.0
3	LSTM	4	0.97	35.0
4	KNN	7	0.97	31.0
5	KNN	3	0.97	19.0
6	MLP	4	0.96	0.0
7	Naive Bayes	7	0.84	5.0
8	Naive Bayes	3	0.50	7.0
9	Naive Bayes	1	0.41	11.0
10	Naive Bayes	2	0.40	11.0
11	KNN	2	0.26	6.0

Los resultados de *KNN* en el escenario 1 fueron los mejores entre todos los clasificadores que se ejecutaron sobre el lavavajillas.

En la Tabla 6.93 se detallan los resultados obtenidos por todos los clasificadores sobre el microondas.

Tabla 6.93: Ranking de resultados obtenidos por los clasificadores en el microondas.

Posición	Clasificador	Escenario	F1-Score	Porcentaje acierto cambios de estado
1	LSTM	7	0.78	40.0
2	KNN	1	0.73	41.0
3	LSTM	4	0.81	28.0
4	KNN	3	0.81	24.0
5	KNN	7	0.67	22.0
6	MLP	4	0.72	0.0
7	Naive Bayes	7	0.29	29.0
8	KNN	2	0.30	19.0
9	Naive Bayes	1	0.10	48.0
10	Naive Bayes	3	0.10	28.0
11	Naive Bayes	2	0.06	50.0

Las redes LSTM con las características del escenario 7 logró los mejores resultados entre todos los obtenidos para el microondas. Se puede observar el gran impacto que tuvo la incorporación de la característica de variación de consumo. Esto aumentó en un 12 % la cantidad de cambios de estados acertados respecto al escenario 4.

La Tabla 6.94 muestra un ranking de los clasificadores en base a sus desempeños en cuanto a *f1-score* y porcentaje de aciertos en cambios de estado sobre la heladera.

Tabla 6.94: Ranking de resultados obtenidos por los clasificadores en la heladera.

Posición	Calsificador	Escenario	F1-Score	Porcentaje acierto cambios de estado
1	LSTM	7	0.99	71.0
2	LSTM	4	0.99	63.0
3	KNN	1	0.97	55.0
4	KNN	7	0.98	49.0
5	KNN	3	0.98	15.0
6	Naive Bayes	7	0.98	0.0
7	Naive Bayes	3	0.98	0.0
8	MLP	4	0.98	0.0
9	Naive Bayes	1	0.96	0.0
10	KNN	2	0.60	37.0
11	Naive Bayes	2	0.08	2.0

La red LSTM logró las mejores métricas alcanzando un *f1-score* de *0.99* y un porcentaje de acierto en los cambios de estado del *71 %*.
La heladera por su parte es el electrodoméstico sobre el que se obtuvieron mejores resultados en general. La principal cualidad de este electrodoméstico es que la activación del mismo no depende de la acción humana, sino que tiene ciclos de ejecución independientes que resultan más fácil de aprender por los clasificadores implementados.

La Tabla 6.95 muestra un ranking de los resultados obtenidos por los distintos clasificadores para la jarra eléctrica.

Tabla 6.95: Ranking de resultados obtenidos por los clasificadores en la jarra eléctrica.

Posición	Clasificador	Escenario	F1-Score	Porcentaje acierto cambios de estado
1	LSTM	7	0.92	75.0
2	LSTM	4	0.91	69.0
3	KNN	7	0.84	53.0
4	KNN	3	0.79	38.0
5	KNN	1	0.68	47.0
6	Naive Bayes	7	0.47	41.0
7	Naive Bayes	1	0.28	75.0
8	Naive Bayes	2	0.28	75.0
9	Naive Bayes	3	0.29	67.0
10	KNN	2	0.25	20.0
11	MLP	4	0.0	0.0

Los mejores resultados del análisis experimental sobre la jarra eléctrica fueron logrados por el algoritmo LSTM, con un *f1-score* de *0.92* y un porcentaje de acierto de cambio de estados del *75 %*, siendo esta última la mejor tasa de toda la evaluación experimental.

La Tabla 6.96 ilustra un ranking basado en el desempeño de los distintos clasificadores sobre la métrica *f1-score* y el porcentaje de aciertos en cambios de estado sobre el lavarropas.

Tabla 6.96: Ranking de resultados obtenidos por los clasificadores en el lavarropas.

Posición	Clasificador	Escenario	F1-Score	Porcentaje acierto cambios de estado
1	LSTM	7	0.91	31.0
2	LSTM	4	0.90	24.0
3	KNN	7	0.88	21.0
4	KNN	3	0.85	11.0
5	KNN	1	0.78	23.0
6	MLP	4	0.86	0.0
7	Naive Bayes	7	0.64	2.0
8	Naive Bayes	3	0.36	3.0
9	Naive Bayes	1	0.33	4.0
10	Naive Bayes	2	0.29	4.0
11	KNN	2	0.19	9.0

El clasificador que registró los valores más significativos fue LSTM con un *f1-score* de *0.91* y un porcentaje de acierto de cambio de estados de *31 %*.
El lavarropas fue el electrodoméstico que presentó mayor dificultad para predecir sus encendidos y apagados. Este inconveniente se puede atribuir a que el lavarropas una vez encendida, durante el lavado y hasta su apagado, funciona en varios niveles de consumo. Por lo tanto no es un electrodoméstico adecuado para la variante binaria del problema de desagregación energética.
A nivel de clasificadores se puede observar que los mejores resultados fueron obtenidos por las redes LSTM. Respecto a la comparación entre los escenarios se observaron varios aspectos interesantes. El primer punto a destacar es que aquellos escenarios en donde se discretizó el tiempo se obtuvieron mayores valores de *f1-score*, lo que se puede atribuir a contar con mayor cantidad de muestras de entrenamiento. Por otra parte, en el escenario 1, donde no se consideró el tiempo en intervalos, se lograron mejores porcentajes de acierto en cambios de estado. Estos porcentajes se pueden explicar debido a que se utilizan valores exactos de consumo durante el entrenamiento y esto conlleva

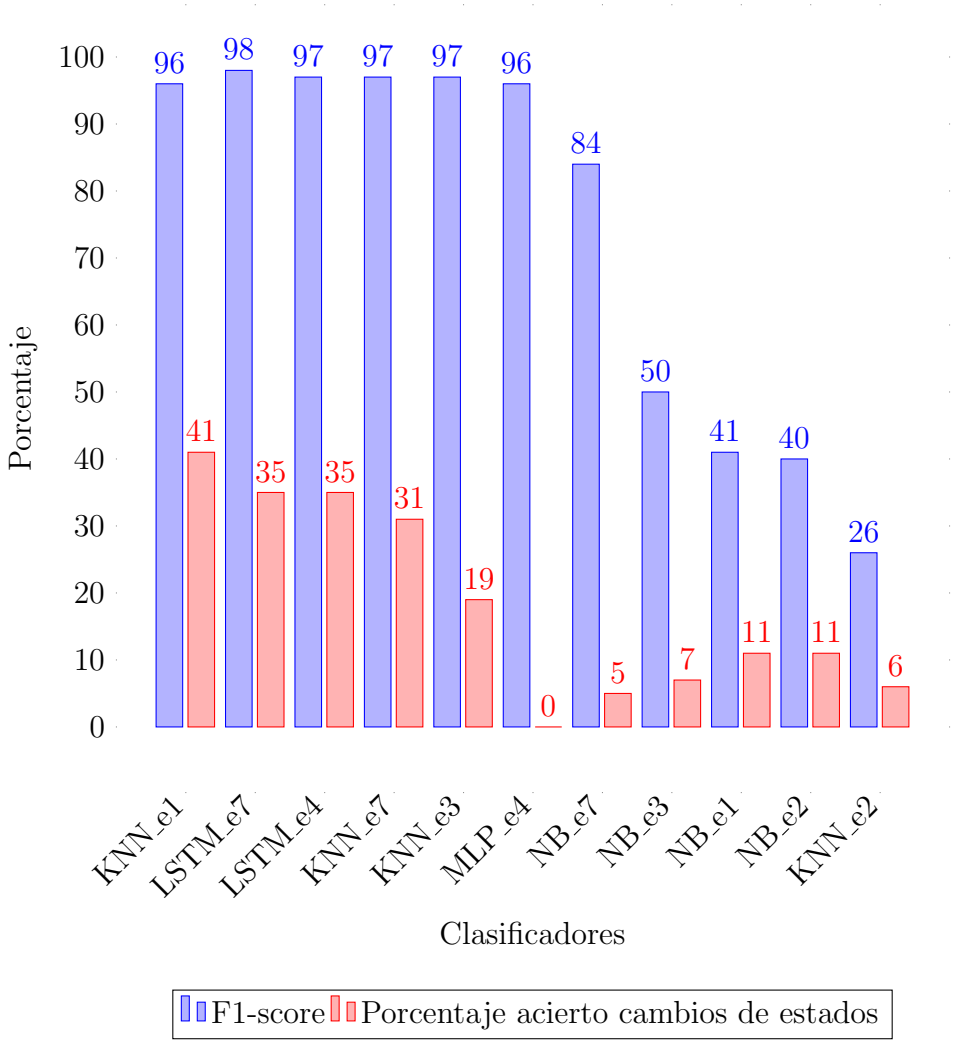
mejor desempeño en la detección de variaciones de consumo. Las conclusiones anteriores se basan en los resultados de *Naive Bayes* y *K Nearest Kneighbors*, que fueron los únicos algoritmos que ejecutaron sobre el escenario 1, ya que por la propia naturaleza de las redes neuronales, entrenar con la baja cantidad de muestras que implica el escenario 1 no es viable.

Un último aspecto a destacar es que la incorporación de la característica de variación de consumo mejoró la detección de cambios de estados sin empeorar el rendimiento general de los clasificadores sobre las métricas de desempeño.

Para finalizar, con el fin de presentar los resultados de forma resumida e ilustrativa se muestran un conjunto de gráficas donde se exponen los resultados logrados por cada uno de los clasificadores implementados, agrupados por electrodoméstico.

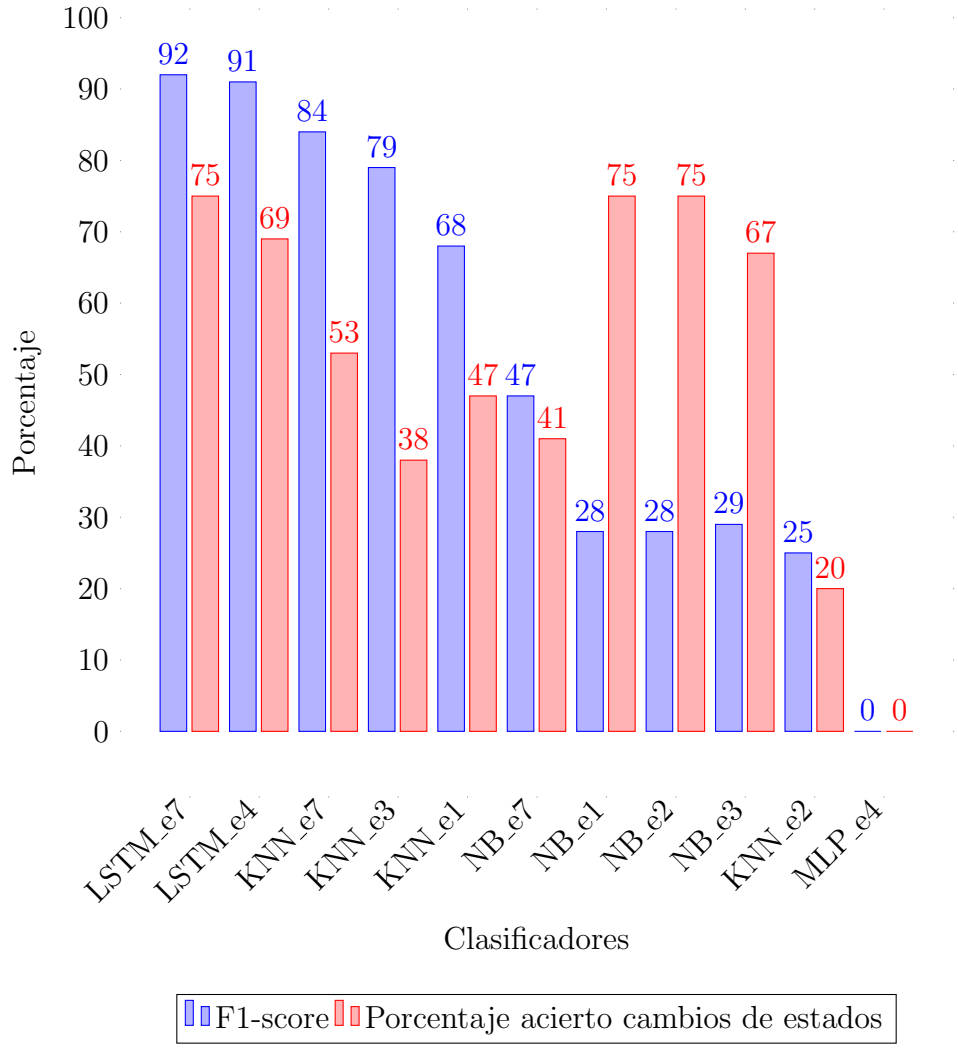
La Figura 6.2 muestra los resultados obtenidos por cada uno de los clasificadores implementados para el lavavajillas.

Figura 6.2: Resultados obtenidos por los clasificadores para el lavavajillas.



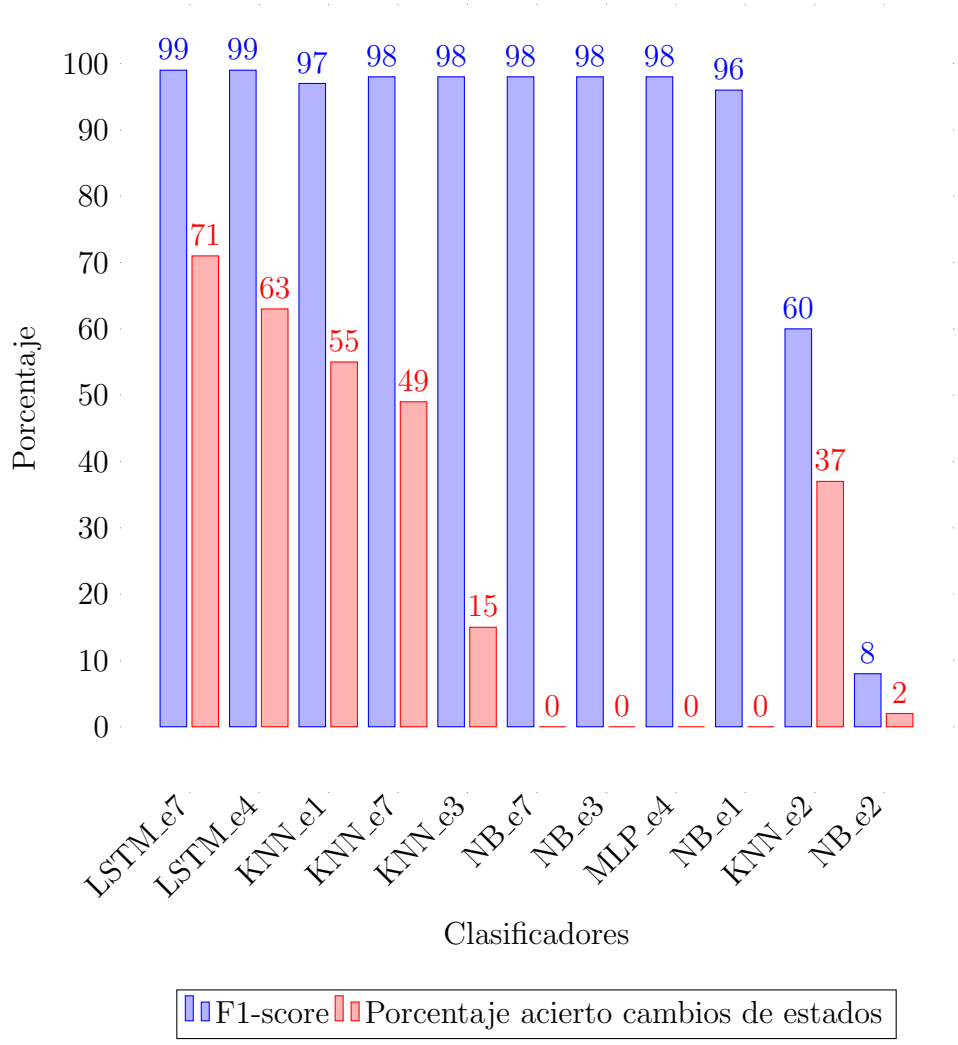
La Figura 6.3 muestra los resultados obtenidos por cada uno de los clasificadores implementados para la jarra eléctrica.

Figura 6.3: Resultados obtenidos por los clasificadores para la jarra eléctrica.



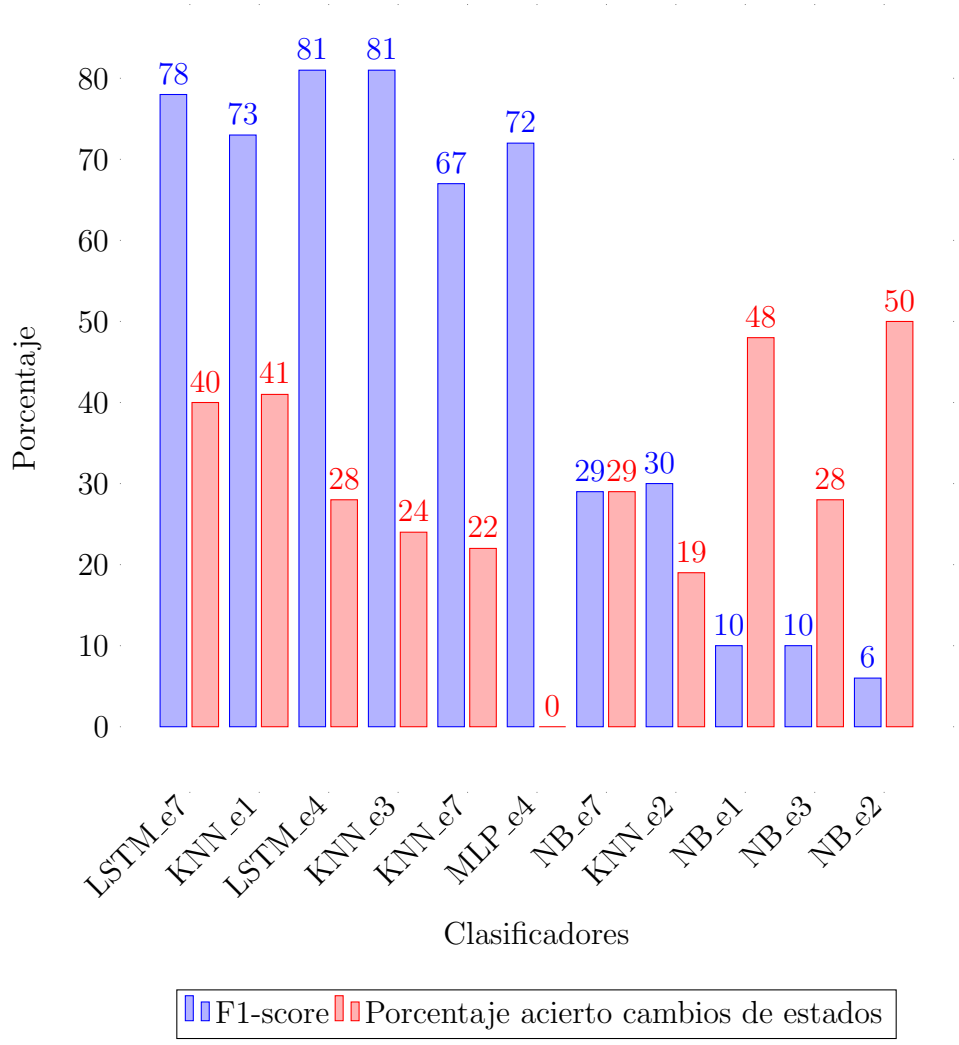
En la Figura 6.8 se ilustran los resultados alcanzados por cada clasificador para la heladera.

Figura 6.4: Resultados obtenidos por los clasificadores para la heladera.



La Figura 6.5 expone los indicadores más relevantes para la evaluación del desempeño de los clasificadores para el microondas.

Figura 6.5: Resultados obtenidos por los clasificadores para el microondas.



Los resultados obtenidos por cada uno de los clasificadores implementados para el lavarropas, se observan en la Figura 6.6.

Figura 6.6: Resultados obtenidos por los clasificadores para el lavarropas.

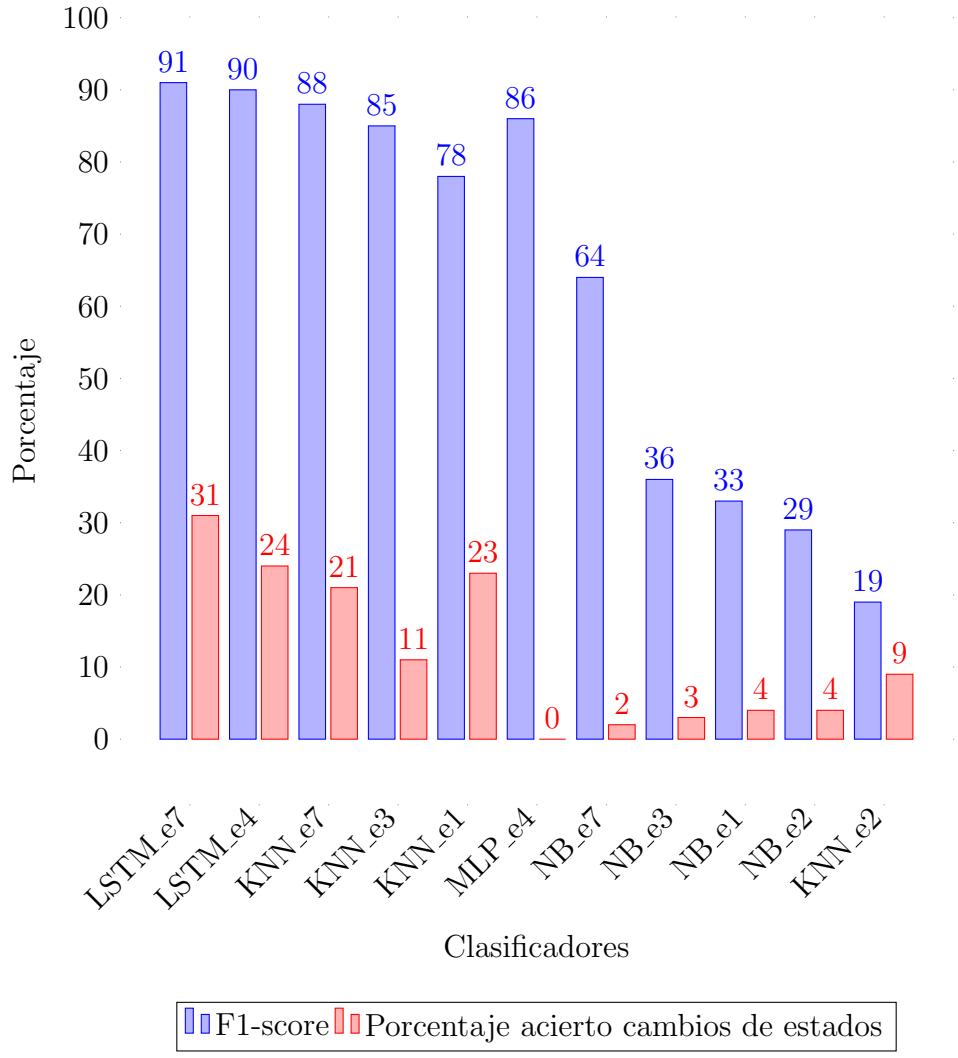


Figura 6.7: Resultados para hogares no vistos en entrenamiento obtenidos por los clasificadores para la heladera.

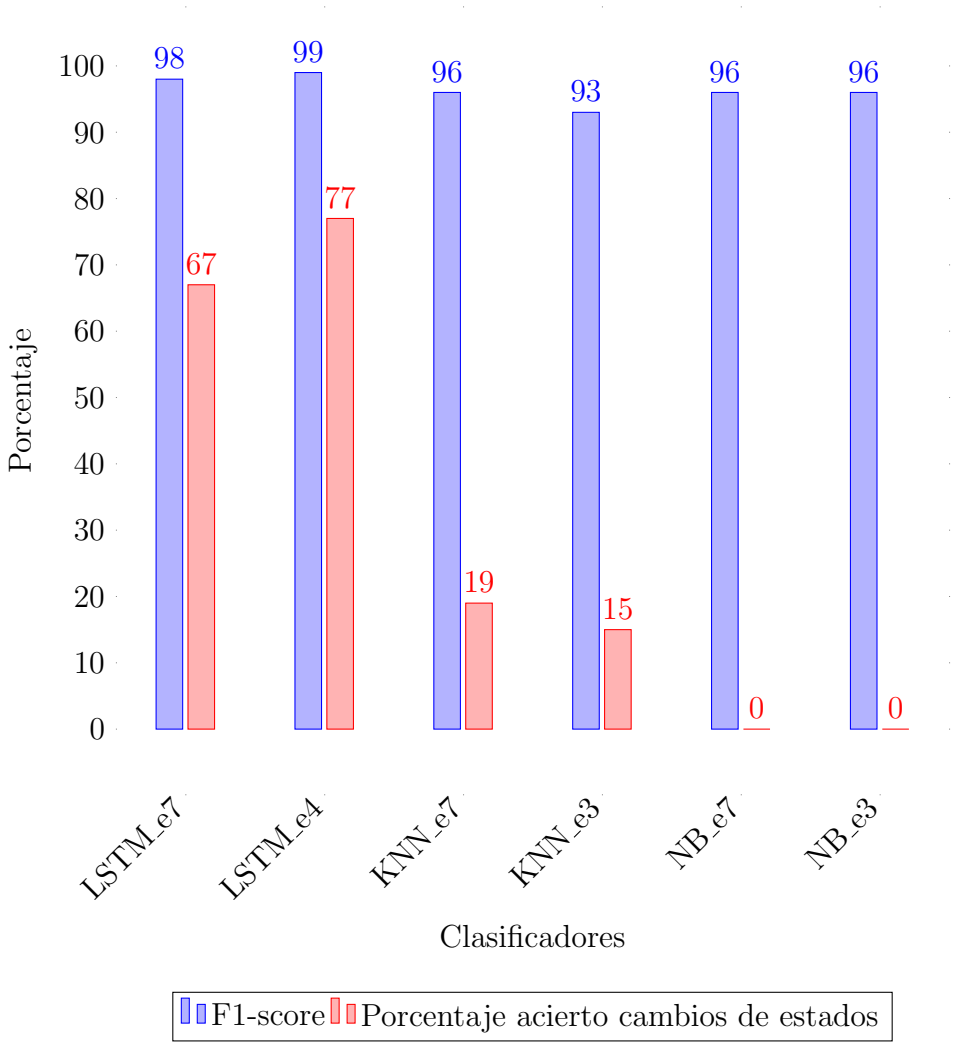
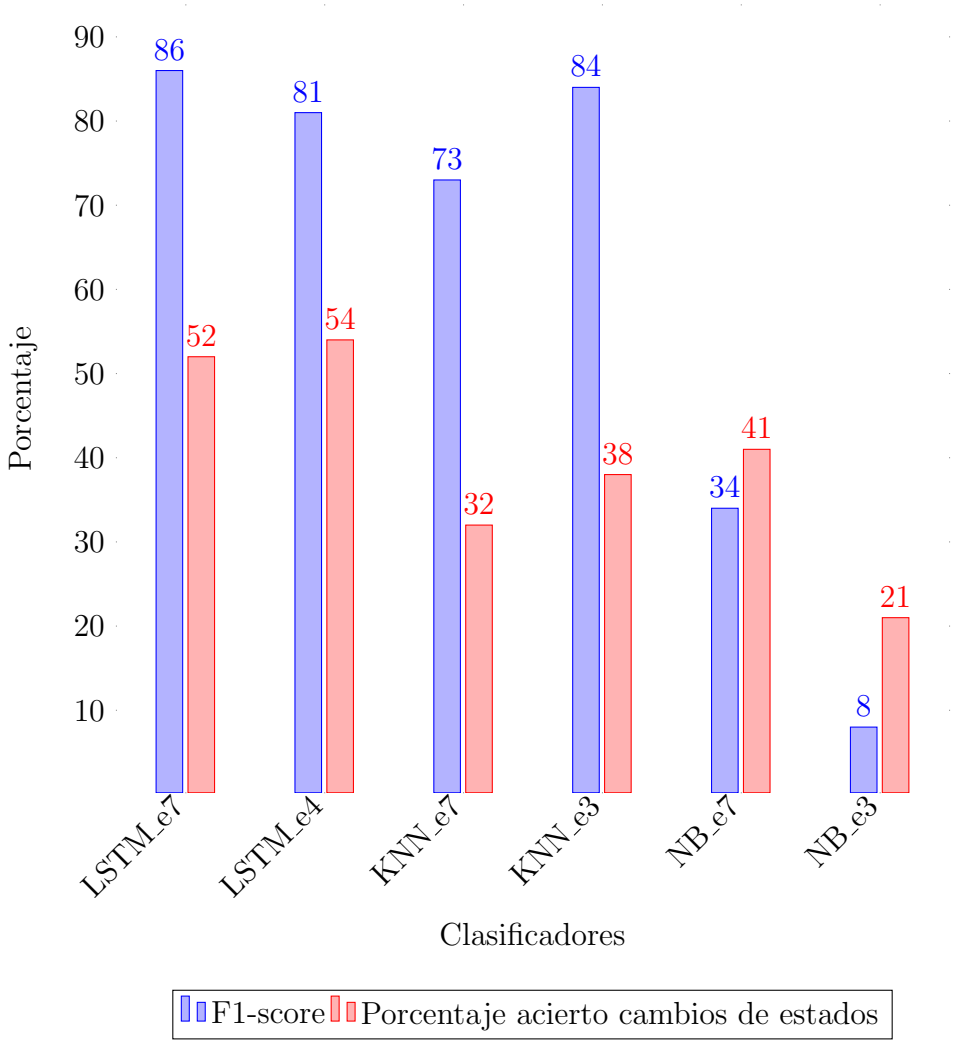


Figura 6.8: Resultados para hogares no vistos en entrenamiento obtenidos por los clasificadores para la microondas.



Queda de manifiesto que las redes con memoria demostraron en sus resultados ser capaces de identificar de forma más precisa las variaciones de consumo características de cada uno de los electrodomésticos. Estos buenos resultados se evidencian tanto para la evaluación en donde las conductas de consumo eran conocidas como para la evaluación en la que no lo eran.

Capítulo 7

Conclusiones y trabajo futuro

En este capítulo se exponen todas las conclusiones arribadas luego de dar por concluido el proceso de trabajo, teniendo como referencia los objetivos planteados al comienzo. Para finalizar se presentan los principales lineamentos del trabajo futuro.

7.1. Conclusiones

Este proyecto abordó el problema de la desagregación energética utilizando algoritmos de aprendizaje automático para su resolución. Con el fin de ahondar en la temática de la desagregación energética, al comienzo del proyecto se realizó un estudió del problema y se investigaron los principales artículos y trabajos de la literatura relacionada.
Se presentaron dos variantes del problema. La línea central del proyecto de grado se desarrolló estudiando la variante binaria, en la cual los electrodomésticos se pueden encontrar únicamente en dos estados, encendido o apagado. La segunda variante consideró múltiples estados de consumo en los electrodomésticos.
Se implementaron cuatro clasificadores: *Naive Bayes, K Nearest Neighbours, redes de perceptrón multicapa y redes Long Short Term Memory.* El entrenamiento y posterior evaluación de los clasificadores se apoyó en un conjunto de datos extraídos del conjunto de datos *UK-DALE.* Se experimentó sobre datos extraídos de dos hogares del Reino Unido comprendidos en el período entre 2012 y 2017.
El desempeño de los clasificadores fue evaluado haciendo énfasis en la métrica

f1-score y en la capacidad para predecir correctamente los cambios de estado en un electrodoméstico. Teniendo en cuenta estos indicadores, las redes LSTM lograron los mejores resultados a nivel general. Los clasificadores de este tipo obtuvieron el mejor desempeño a la hora de predecir el comportamiento de la heladera alcanzando, en el mejor escenario, un valor de 0.99 para la métrica f1-score y un porcentaje de acierto en los cambios de estado del 71 %. Por otro lado, estos clasificadores tuvieron los mayores inconvenientes a la hora de predecir el comportamiento del microondas dónde en el mejor escenario lograron un 40 % en porcentaje de acierto en cambios de estado y 0.71 para la métrica f1-score. Se constató que considerar la variación de consumo agregado aumentó el porcentaje de acierto en cambios de estado. Por otra parte, trabajar con intervalos de tiempo discretizado elevó los valores obtenidos en las métricas *f1-score, precision, accuracy, recall*, pero a su vez disminuyó la capacidad de los clasificadores para detectar cambios de estado.
El último aspecto trabajado en este proyecto fue la evaluación de los métodos sobre datos de hogares no vistos durante el entrenamiento, con el objetivo de comprobar la capacidad de los clasificadores para adaptarse a diferentes electrodomésticos de un mismo tipo y a diferentes patrones de consumo. Los resultados obtenidos fueron similares en la heladera e incluso mejores para el microondas. Para este último electrodoméstico, las redes LSTM logran aumentar un 12 % la tasa de acierto en cambios de estados tanto para el escenario 4 como para el escenario 7. Las diferencias en la métrica f1-score por su parte no fueron significativas.

7.2. Trabajo futuro

Esta sección describe las principales líneas de trabajo futuro que surgieron durante el desarrollo del proyecto de grado.
Durante la investigación se trabajó sobre datos disponibles en el repositorio UK-DALE. Una de las principales tareas a futuro es trabajar con datos provenientes de hogares uruguayos, con el objetivo de brindarle a los ciudadanos sugerencias para el uso de sus electrodomésticos, lograr una reducción en el consumo eléctrico y consecuentemente un beneficio económico.
Por otro lado, el proyecto de grado se enfocó sobre cinco electrodomésticos particulares. Es necesario extender el estudio sobre la mayor cantidad de electrodomésticos posible, ya que cada uno tiene sus propias características y los

métodos desarrollados deben entrenarse de forma particular para cada uno de ellos.

Otra línea de trabajo futuro incluye ahondar en el estudio de las redes neuronales. Investigar nuevas variantes de redes neuronales o arquitecturas más complejas que logren mejorar los resultados obtenidos. En particular, intentar superar los resultados en aquellos electrodomésticos que presentan patrones de uso poco predecibles.

Finalmente, además de trabajar con la variante binaria del problema de desagregación energética, es deseable desarrollar nuevos clasificadores que aborden la variante de múltiples estados de consumo para tener un enfoque más adecuado a la realidad.

Referencias bibliográficas

[1] UTE, "Plan estratégico 2019." https://portal.ute.com.uy/sites/default/files/files-cuerpo-paginas/Plan%20Estrat%C3%A9gico%20UTE%202019%20-%20Transparencia%20UTEa_0.pdf. Online; accedido en Junio 2019.

[2] G. Hart, "Nonintrusive appliance load monitoring," *Proceedings of the IEEE*, vol. 80, no. 12, pp. 1870–1891, 1992.

[3] A. Faustine, N. H. Mvungi, S. Kaijage, and K. Michae, "A survey on non-intrusive load monitoring methodies and techniques for energy disaggregation problem," 2017.

[4] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[5] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *International Conference on Learning Representations*, 2014.

[6] T. Mitchell, *Machine Learning*. McGraw-Hill, 1997.

[7] J. Kelly, *Disaggregation of Domestic Smart Meter Energy Data*. Imperial College London, 2016.

[8] T. Oliphant, "NumPy: A guide to NumPy." USA: Trelgol Publishing, 2006.

[9] W. McKinney, "Data structures for statistical computing in python," in *Proceedings of the 9th Python in Science Conference* (S. van der Walt and J. Millman, eds.), pp. 51 – 56, 2010.

[10] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[11] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. Software available from tensorflow.org.

[12] F. Chollet, "Keras." https://github.com/fchollet/keras, 2015.

[13] Theano Development Team, "Theano: A Python framework for fast computation of mathematical expressions," *arXiv e-prints*, vol. abs/1605.02688, 2016.

[14] J. Kolter and M. Johnson, "REDD: A public data set for energy disaggregation research," *SustKDD workshop on Data Mining Applications in Sustainability*, 2011.

[15] J. Kelly and W. Knottenbelt, "The UK-DALE dataset, domestic appliance-level electricity demand and whole-house demand from five UK homes," *Scientific Data*, vol. 2, no. 150007, 2015.

[16] S. Nesmachnow and S. Iturriaga, "Cluster-UY: High Performance Scientific Computing in Uruguay," *International Supercomputing Conference in Mexico*, 2019.

APÉNDICES

Apéndice 1

Resultados de clase apagado para la variante binaria

1.1. Naive Bayes

Tabla 1.1: Resultados de Naive Bayes en el escenario 1 para la clase apagado.

Clase apagado				
Electrodoméstico	TP	FP	TN	FN
Lavavajillas	517642	9878	5864	7105
Microondas	476700	244	879	15480
Heladera	304194	8167	220094	8253
Jarra eléctrica	522172	154	1461	7357
Lavarropas	490525	9310	5170	11729

Tabla 1.2: Métricas de Naive Bayes en el escenario 1 para la clase apagado.

Clase apagado			
Electrodoméstico	F1 Score	Precision	Recall
Lavavajillas	0,9839	0,9813	0,9864
Microondas	0,9838	0,9995	0,9686
Heladera	0,9737	0,9739	0,9736
Jarra eléctrica	0,9929	0,9997	0,9861
Lavarropas	0,9790	0,9814	0,9767

Tabla 1.3: Resultados de Naive Bayes en el escenario 2 para la clase apagado.

Clase apagado				
Electrodoméstico	TP	FP	TN	FN
Lavavajillas	516823	9878	5864	7924
Microondas	472444	444	679	19736
Heladera	306223	218918	9343	6224
Jarra eléctrica	522173	152	1463	7356
Lavarropas	486690	9293	5187	15564

Tabla 1.4: Métricas de Naive Bayes en el escenario 2 para la clase apagado.

Clase apagado			
Electrodoméstico	F1 Score	Precision	Recall
Lavavajillas	0,9831	0,9813	0,9849
Microondas	0,9791	0,9991	0,9599
Heladera	0,7312	0,5831	0,9801
Jarra eléctrica	0,9929	0,9997	0,9861
Lavarropas	0,9751	0,9813	0,9690

Tabla 1.5: Resultados de Naive Bayes en el escenario 3 para la clase apagado.

Clase apagado				
Electrodoméstico	TP	FP	TN	FN
Lavavajillas	843287	16424	15510	14052
Microondas	854427	754	1546	26383
Heladera	498248	7564	367332	7565
Jarra eléctrica	836115	789	5722	27955
Lavarropas	840384	17206	9140	15528

Tabla 1.6: Métricas de Naive Bayes en el escenario 3 para la clase apagado.

Clase apagado			
Electrodoméstico	F1 Score	Precision	Recall
Lavavajillas	0,9823	0,9809	0,9836
Microondas	0,9844	0,9991	0,9700
Heladera	0,9850	0,9851	0,9850
Jarra eléctrica	0,9831	0,9991	0,9677
Lavarropas	0,9809	0,9799	0,9819

Tabla 1.7: Resultados de Naive Bayes en el escenario 7 para la clase apagado.

Clase apagado				
Electrodoméstico	TP	FP	TN	FN
Lavavajillas	850346	3868	28066	6993
Microondas	872381	495	1805	8429
Heladera	498194	7562	367334	7619
Jarra eléctrica	855876	2044	4467	8194
Lavarropas	848750	10643	15703	7162

Tabla 1.8: Métricas de Naive Bayes en el escenario 7 para la clase apagado.

Clase apagado			
Electrodoméstico	F1 Score	Precision	Recall
Lavavajillas	0,9937	0,9955	0,9918
Microondas	0,9949	0,9994	0,9904
Heladera	0,9850	0,9851	0,9849
Jarra eléctrica	0,9941	0,9976	0,99051
Lavarropas	0,9896	0,9876	0,9916

1.2. K Nearest Neighbours

Tabla 1.9: Resultados de KNN en el escenario 1 para la clase apagado.

Clase apagado				
Electrodoméstico	TP	FP	TN	FN
Lavavajillas	262015	275	7589	365
Microondas	245987	175	382	107
Heladera	152946	4175	109942	3291
Jarra eléctrica	264625	313	486	148
Lavarropas	249064	1320	5944	2039

Tabla 1.10: Métricas de KNN en el escenario 1 para la clase apagado.

Clase apagado			
Electrodoméstico	F1 Score	Precision	Recall
Lavavajillas	0,9987	0,9989	0,9986
Microondas	0,9993	0,9992	0,9995
Heladera	0,9761	0,9734	0,9789
Jarra eléctrica	0,999	0,9988	0,9994
Lavarropas	0,9932	0,9947	0,9918

Tabla 1.11: Resultados de KNN en el escenario 2 para la clase apagado.

Clase apagado				
Electrodoméstico	TP	FP	TN	FN
Lavavajillas	261548	6564	1300	832
Microondas	245859	419	138	235
Heladera	103675	43902	70215	52562
Jarra eléctrica	264492	647	152	281
Lavarropas	246278	5968	1296	4825

Tabla 1.12: Métricas de KNN en el escenario 2 para la clase apagado.

Clase apagado			
Electrodoméstico	F1 Score	Precision	Recall
Lavavajillas	0,986	0,9755	0,9968
Microondas	0,9985	0,9982	0,999
Heladera	0,6824	0,7025	0,6635
Jarra eléctrica	0,9981	0,9975	0,9989
Lavarropas	0,9784	0,9763	0,9807

Tabla 1.13: Resultados de KNN en el escenario 3 para la clase apagado.

Clase apagado				
Electrodoméstico	TP	FP	TN	FN
Lavavajillas	428040	432	15521	643
Microondas	440051	370	968	166
Heladera	260191	3545	172235	4384
Jarra eléctrica	431297	921	2617	456
Lavarropas	426129	1372	11316	2312

Tabla 1.14: Métricas de KNN en el escenario 3 para la clase apagado.

Clase apagado			
Electrodoméstico	F1 Score	Precision	Recall
Lavavajillas	0,9986	0,9989	0,9985
Microondas	0,9993	0,9991	0,9996
Heladera	0,9849	0,9865	0,9834
Jarra eléctrica	0,9983	0,9978	0,9989
Lavarropas	0,9956	0,9967	0,9946

Tabla 1.15: Resultados de KNN en el escenario 7 para la clase apagado.

Clase apagado				
Electrodoméstico	TP	FP	TN	FN
Lavavajillas	425877	436	17733	591
Microondas	439816	474	864	401
Heladera	261546	2698	173082	3029
Jarra eléctrica	431281	633	2905	472
Lavarropas	426547	1328	11360	1894

Tabla 1.16: Métricas de KNN en el escenario 7 para la clase apagado.

Clase apagado			
Electrodoméstico	F1 Score	Precision	Recall
Lavavajillas	0,9987	0,9989	0,9986
Microondas	0,9989	0,9989	0,999
Heladera	0,989	0,9897	0,9885
Jarra eléctrica	0,9986	0,9985	0,9989
Lavarropas	0,9961	0,9968	0,9955

1.3. Multi Layer Perceptron

Tabla 1.17: Resultados de MLP en el escenario 4 para la clase apagado.

Clase apagado				
Electrodoméstico	TP	FP	TN	FN
Lavavajillas	425798	670	17499	670
Microondas	439848	369	969	369
Heladera	260896	3679	172101	3679
Jarra eléctrica	431753	3538	0	0
Lavarropas	426705	1736	10952	1736

Tabla 1.18: Métricas de MLP en el escenario 4 para la clase apagado.

Clase apagado			
Electrodoméstico	F1 Score	Precision	Recall
Lavavajillas	0,9984	0,9984	0,9984
Microondas	0,9991	0,9991	0,9991
Heladera	0,986	0,986	0,986
Jarra eléctrica	9,9958	0,9918	1
Lavarropas	0,9959	0,9959	0,9959

1.4. Long Short Term Memory

Tabla 1.19: Resultados de redes LSTM en el escenario 4 para la clase apagado.

Clase apagado				
Electrodoméstico	TP	FP	TN	FN
Lavavajillas	426015	488	17681	446
Microondas	440134	382	956	78
Heladera	262330	1700	174080	2240
Jarra eléctrica	431582	446	3092	169
Lavarropas	426818	986	11702	1620

Tabla 1.20: Métricas de redes LSTM en el escenario 4 para la clase apagado.

Clase apagado			
Electrodoméstico	F1 Score	Precision	Recall
Lavavajillas	0,9989	0,9989	0,9990
Microondas	0,9995	0,9991	0,9998
Heladera	0,9925	0,9937	0,9915
Jarra eléctrica	0,9993	0,9990	0,9996
Lavarropas	0,9970	0,9977	0,9962

Tabla 1.21: Resultados de redes LSTM en el escenario 7 para la clase apagado.

Clase apagado				
Electrodoméstico	TP	FP	TN	FN
Lavavajillas	426218	604	17565	246
Microondas	439955	312	1026	257
Heladera	262387	2042	173738	2185
Jarra eléctrica	431545	359	3179	206
Lavarropas	427117	914	11774	1305

Tabla 1.22: Métricas de redes LSTM en el escenario 7 para la clase apagado.

Clase apagado			
Electrodoméstico	F1 Score	Precision	Recall
Lavavajillas	0,9990	0,9986	0,9994
Microondas	0,9994	0,9993	0,9994
Heladera	0,9920	0,9923	0,9917
Jarra eléctrica	0,9992	0,9976	0,9995
Lavarropas	0,9974	0,9979	0,9970