



ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO
FACULTAD DE MECÁNICA
CARRERA MANTENIMIENTO INDUSTRIAL

**“ANÁLISIS DE EFICIENCIA ENERGÉTICA EN EDIFICIOS
RESIDENCIALES UTILIZANDO APRENDIZAJE DE MÁQUINAS
BASADO EN REGRESIÓN LINEAL”**

Trabajo de Integración Curricular

Tipo: Proyecto de Investigación

Presentado para optar al grado académico de:

INGENIERO EN MANTENIMIENTO INDUSTRIAL

AUTOR:

JAVIER AMADOR CHARANCHI SALTOS

Riobamba-Ecuador

2022



ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO
FACULTAD DE MECÁNICA
CARRERA MANTENIMIENTO INDUSTRIAL

**“ANÁLISIS DE EFICIENCIA ENERGÉTICA EN EDIFICIOS
RESIDENCIALES UTILIZANDO APRENDIZAJE DE MÁQUINAS
BASADO EN REGRESIÓN LINEAL”**

Trabajo de Integración Curricular

Tipo: Proyecto de Investigación

Presentado para optar al grado académico de:

INGENIERO EN MANTENIMIENTO INDUSTRIAL

AUTOR: JAVIER AMADOR CHARANCHI SALTOS

DIRECTOR: Ing. FÉLIX ANTONIO GARCÍA MORA

Riobamba-Ecuador

2022

© 2022, Javier Amador Charanchi Saltos

Se autoriza la reproducción total o parcial, con fines académicos, por cualquier medio o procedimiento, incluyendo cita bibliográfica del documento, siempre y cuando se reconozca el Derecho de Autor.

Yo, JAVIER AMADOR CHARANCHI SALTOS, declaro que el presente trabajo de integración curricular es de mi autoría y los resultados del mismo son auténticos. Los textos en el documento que provienen de otras fuentes están debidamente citados y referenciados.

Como autor asumo la responsabilidad legal y académica de los contenidos de este trabajo de integración curricular; el patrimonio intelectual pertenece a la Escuela Superior Politécnica de Chimborazo.

Riobamba, 07 de junio de 2022





Javier Amador Charanchi Saltos

1724069982

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO
FACULTAD DE MECÁNICA
CARRERA MANTENIMIENTO INDUSTRIAL

El Tribunal del Trabajo de Integración Curricular certifica que: El Trabajo de Integración Curricular: Tipo: Proyecto de Investigación, “**ANÁLISIS DE EFICIENCIA ENERGÉTICA EN EDIFICIOS RESIDENCIALES UTILIZANDO APRENDIZAJE DE MÁQUINAS BASADO EN REGRESIÓN LINEAL**”, realizado por el señor: **JAVIER AMADOR CHARANCHI SALTOS**, ha sido minuciosamente revisado por los Miembros del Tribunal del Trabajo de Integración Curricular, el mismo que cumple con los requisitos científicos, técnicos, legales, en tal virtud el Tribunal Autoriza su presentación.

	FIRMA	FECHA
Ing. Marco Antonio Ordóñez Viñán PRESIDENTE DEL TRIBUNAL		2022-06-07
Ing. Félix Antonio García Mora DIRECTOR DE TRABAJO DE INTEGRACIÓN CURRICULAR		2022-06-07
Ing. Eduardo Segundo Hernández Dávila MIEMBRO DEL TRIBUNAL		2022-06-07

DEDICATORIA

El presente trabajo de integración curricular es la culminación de un escalón más en mi vida el cual se lo dedico, en primer lugar, a Dios quien ha guiado toda mi vida y me ha dado las fuerzas necesarias para vencer todos los obstáculos y no rendirme hasta cumplir todos los objetivos que me he propuesto.

A mis padres Luis Charanchi y Marcia Saltos, quienes, con su amor, sabiduría, dedicación y su apoyo incondicional a lo largo de toda mi vida, me han sabido guiar y permitiéndome cumplir un sueño más, gracias a ellos que inculcaron en mí, buenos valores y la enseñanza de que, con esfuerzo y constancia todo se puede lograr en esta vida.

Javier

AGRADECIMIENTO

Agradezco a Dios por guiarme en mi camino, por brindarme la salud, la fuerza y el tiempo necesario para concluir con gran éxito mi carrera profesional y a mi familia quien con su constante motivación y apoyo incondicional me permitieron seguir siempre adelante y no rendirme.

A la Escuela Superior Politécnica de Chimborazo y a la prestigiosa carrera de Mantenimiento Industrial, por haberme acogido durante todo mi proceso de estudiante y darme la oportunidad de tener una excelente formación académica con docentes que supieron guiarme y brindarme su conocimiento, en especial agradezco a mi tutor Ing. Félix García y mi asesor Ing. Eduardo Hernández quienes me con su sabiduría y experiencia permitieron el desarrollo del presente trabajo de integración curricular.

Javier

TABLA DE CONTENIDO

ÍNDICE DE TABLAS.....	x
ÍNDICE DE FIGURAS.....	xi
ÍNDICE DE GRÁFICOS.....	xii
ÍNDICE DE ABREVIATURAS.....	xiii
ÍNDICE DE ANEXOS.....	xiv
RESUMEN.....	xv
SUMMARY.....	¡Error! Marcador no definido.
INTRODUCCIÓN.....	1

CAPÍTULO I

1. MARCO TEÓRICO CONCEPTUAL.....	4
1.1. La eficiencia energética en relación con los edificios residenciales.....	4
1.1.1. Variables para la medición de la eficiencia energética en edificios residenciales.....	4
1.1.2. La eficiencia energética asociada al confort térmico.....	5
1.1.3. Impulsores de valor basados en la eficiencia energética.....	6
1.2. Introducción a Anaconda con Python.....	7
1.2.1. Anaconda.....	7
1.2.2. Python.....	8
1.2.2.1. ¿Por qué aprender Python?.....	8
1.3. Inteligencia artificial.....	9
1.3.1. Importancia de la inteligencia artificial.....	9
1.3.2. Aplicaciones más comunes de la inteligencia artificial.....	10
1.4. Machine learning (ML).....	10
1.4.1. Tipos de aprendizaje en machine learning.....	11
1.4.1.1. Aprendizaje supervisado.....	11
1.4.1.2. Aprendizaje no supervisado.....	12
1.4.1.3. Aprendizaje por refuerzo.....	13
1.4.2. Pasos para construir un modelo de machine learning.....	14
1.4.3. Problemas del machine learning más comunes.....	15
1.5. Principales modelos de machine learning basados en de regresión.....	16
1.5.1. Regresión lineal (RL).....	16
1.5.1.1. Importancia de la regresión lineal.....	17

1.5.2.	<i>Árbol de decisión (AD)</i>	18
1.5.3.	<i>Random Forest Regression (RFR)</i>	18
1.6.	Métricas de evaluación de un modelo de ML basado en regresión	20
1.7.	Detección del consumo de energía utilizando aprendizaje de maquinas	22

CÁPITULO II

2.	MARCO METODOLÓGICO	23
2.1.	Recolección de los datos	23
2.1.1.	<i>Definición del problema</i>	24
2.2.	Preprocesamiento del conjunto de datos	24
2.2.1.	<i>Descripción general del conjunto de datos</i>	25
2.2.2.	<i>Importación de librerías</i>	26
2.2.3.	<i>Lectura del conjunto de datos</i>	27
2.2.4.	<i>Conjunto de datos de eficiencia energética</i>	28
2.2.5.	<i>Limpieza de los datos</i>	28
2.3.	Análisis exploratorio de los datos	29
2.3.1.	<i>Búsqueda de la información del conjunto de datos</i>	29
2.3.2.	<i>Descripción estadística de los datos</i>	30
2.3.3.	<i>Análisis grafico</i>	31
2.3.4.	<i>Análisis de correlación</i>	32
2.4.	Entrenamiento del modelo	34
2.4.1.	<i>Normalizado del conjunto de datos</i>	34
2.4.2.	<i>División del conjunto de datos</i>	34
2.4.3.	<i>Creación del modelo de ML</i>	35
2.5.	Validación del modelo	35
2.5.1.	<i>Predicción del conjunto de datos</i>	35
2.5.2.	<i>Métricas para la evaluación del modelo</i>	36

CAPÍTULO III

3.	RESULTADOS Y DISCUSIÓN	37
3.1.	Interpretación de los resultados obtenidos en cada método experimental	37
3.2.	Modelo experimental 1 basado en Regresión Lineal	37
3.2.1.	<i>Métricas de evaluación del modelo experimental 1</i>	37
3.2.2.	<i>Comparación de los valores del conjunto de datos y los predichos por el modelo</i> ...	39
3.3.	Modelo experimental 2 basado en Decisión Tree Regressor	39

3.3.1.	<i>Métricas de evaluación del modelo experimental 2</i>	40
3.3.2.	<i>Comparación de los valores del conjunto de datos y los predichos por el modelo</i> ...	42
3.3.3.	<i>Optimización del modelo experimental 2</i>	42
3.3.4.	<i>Métricas de evaluación del modelo experimental 2 después de la optimización</i>	44
3.3.5.	<i>Comparación de los valores del conjunto de datos y los predichos por el modelo optimizado</i>	44
3.4.	Modelo experimental 3 basado en Random Forest Regression	45
3.4.1.	<i>Métricas de evaluación del modelo experimental 3</i>	45
3.4.2.	<i>Comparación de los valores del conjunto de datos y los predichos por el modelo</i> ...	47
3.4.3.	<i>Optimización del modelo experimental 3</i>	47
3.4.4.	<i>Métricas de evaluación del modelo experimental 3 después de la optimización</i>	49
3.4.5.	<i>Comparación de los valores del conjunto de datos y los predichos por el modelo optimizado</i>	49
3.5.	Modelo experimental 4 basado en Extra Tree Regression	50
3.5.1.	<i>Métricas de evaluación del modelo experimental 4</i>	50
3.5.2.	<i>Comparación de los valores del conjunto de datos y los predichos por el modelo</i> ...	52
3.5.3.	<i>Optimización del modelo experimental 4</i>	52
3.5.4.	<i>Métricas de evaluación del modelo experimental 4 después de la optimización</i>	54
3.5.5.	<i>Comparación de los valores del conjunto de datos y los predichos por el modelo optimizado</i>	54
3.6.	<i>Comparación de los modelos experimentales</i>	55
CONCLUSIONES		56
RECOMENDACIONES		57
BIBLIOGRAFIA		
ANEXOS		

ÍNDICE DE TABLAS

Tabla 1-1: Variables necesarias para la medición de la eficiencia energética.	5
Tabla 1-2: Conjunto de datos de eficiencia energética.	28
Tabla 2-2: Descripción estadística del conjunto de datos de eficiencia energética.	31

ÍNDICE DE FIGURAS

Figura 1-2: Repositorio de la Universidad de California Irvine (UCI).....	23
Figura 2-2: Software Anaconda navigator.....	25
Figura 3-2: Conjunto de datos de eficiencia energética.....	26
Figura 4-2: Conjunto de datos de eficiencia energética.....	26
Figura 5-2: Descripción estadísticas del conjunto de datos	32
Figura 6-2: Mapa de correlaciones del conjunto de datos	33
Figura 1-3: Precisión del modelo experimental 1 basado en	39
Figura 2-3: Comparación de los valores predichos del modelo experimental 1	39
Figura 3-3: Precisión del modelo experimental 2	41
Figura 4-3: Comparación de los valores predichos del modelo experimental 2	42
Figura 5-3: Mejores parámetros del modelo experimental 2	43
Figura 6-3: Comparación de los valores predichos del modelo experimental 2 optimizado	44
Figura 7-3: Precisión del modelo experimental 3	46
Figura 8-3: Comparación de los valores predichos del modelo experimental 3	47
Figura 9-3: Mejores parámetros del modelo experimental 3	48
Figura 10-3: Comparación de los valores predichos del modelo experimental 3 optimizado ...	49
Figura 11-3: Precisión del modelo experimental 4	51
Figura 12-3: Comparación de los valores predichos del modelo experimental 4	52
Figura 13-3: Mejores parámetros del modelo experimental 4	53
Figura 14-3: Comparación de los valores predichos del modelo experimental 4 optimizado ...	54
Figura 15-3: Comparación de los modelos de predicción	55

ÍNDICE DE GRÁFICOS

Gráfico 1-1: Beneficios de la eficiencia energética.	4
Gráfico 2-1: Consumo de energía de un edificio residencial	6
Gráfico 3-1: Tipos de machine learning	11
Gráfico 4-1: Flujo de trabajo del aprendizaje supervisado	12
Gráfico 5-1: Flujo de trabajo del aprendizaje no supervisado	13
Gráfico 6-1: Flujo de trabajo del aprendizaje por refuerzo.....	13
Gráfico 7-1: Proceso de construcción del modelo de aprendizaje automático	14
Gráfico 8-1: Modelos de machine learning más utilizados en regresión.	16
Gráfico 9-1: Representación de un proceso de regresión lineal.....	17
Gráfico 10-1: Proceso de construcción de un modelo de Random Forest.	20

ÍNDICE DE ABREVIATURAS

PE	Pobreza Energética
CC	Carga de calefacción.
CR	Carga de refrigeración.
REE	Rendimiento energético de edificios
IA	Inteligencia artificial.
ML	Aprendizaje de máquinas.

ÍNDICE DE ANEXOS

ANEXO A: PROGRAMACIÓN FINALIZADA

RESUMEN

El presente trabajo de integración curricular tiene como objetivo principal aplicar técnicas de análisis de aprendizaje automático (ML) basado en métodos de regresión para predecir las cargas de calefacción y refrigeración en edificios residenciales, a través de la generación de ecuaciones de regresión y código en lenguaje de Python para las técnicas de ML, por lo que, para la creación del algoritmo de aprendizaje se utilizó un data set de rendimiento energético cuya base de datos fue extraída del Machine Learning Repository de la Universidad de California, Irvine, este conjunto de datos contiene ocho atributos (o características, denotados por X1 a X8) y dos respuestas (o resultados, denotados por Y1 y Y2) cuya finalidad es utilizar las ocho funciones para predecir cada una de las dos respuestas. La primera parte del trabajo está dedicada al análisis exploratorio de datos y visualizaciones, así como a la aplicación de técnicas estadísticas y análisis de componentes principales del data set. Luego de este análisis en una segunda parte, se desarrollaron varios métodos de regresión, donde se comparará los rendimientos de cada algoritmo, con el objetivo de encontrar el modelo más óptimo para predecir la eficiencia energética en los edificios residenciales, por último se evaluarán los diferentes algoritmos de regresión, analizando los resultados, la precisión y los errores que arrojan estos, donde se determinó que los modelos de Random Forest Regression, Decision Tree Regressor, Extra Tree Regressor y el modelo de regresión lineal son modelos aplicables a la predicción del valor de la carga de calefacción y refrigeración, debido a que los cuatro modelos muestran precisiones mayores al 90%, siendo el modelo de Random forest el más adecuado para este estudio por dar una precisión del 98%, Se recomienda realizar otros modelos de ML con la finalidad de mejorar los resultados de las predicciones.

Palabras clave: <INTELIGENCIA ARTIFICIAL>, <EFICIENCIA ENERGÉTICA>, <APRENDIZAJE DE MÁQUINAS>, <REGRESIÓN>, <EDIFICIOS RESIDENCIALES>, <RENDIMIENTO ENERGÉTICO>.



Elaborado y digitalizado por:
**HOLGER GERMAN
RAMOS UVIDIA**

1331-DBRA-UTP-2022

2022-06-29

SUMMARY

The objective of this curricular integration work was to apply automatic learning analysis techniques (ML). It was based on regression methods to predict heating and cooling loads in residential buildings, through the generation of regression equations and Python language code for ML techniques. Therefore, for the creation of the learning algorithm, an energy performance data set was used whose database was extracted from the Machine Learning Repository of California University, Irvine. This data set contains eight attributes (or characteristics, denoted by X1 through X8) and two responses (or outcomes, denoted Y1 and Y2) whose purpose is to use the eight functions to predict each of the two responses. The first part of the work is dedicated to the exploratory analysis of data and visualizations, as well as the application of statistical techniques and analysis of the main components of the data set. After this analysis in a second part, several regression methods were developed, where the performance of each algorithm was compared, with the aim of finding the most optimal model to predict energy efficiency in residential buildings. Finally the different regression algorithms, analyzing the results, the precision and the errors that these showed. The Random Forest Regression, Decision Tree Regressor, Extra Tree Regressor and the linear regression model were determined that are models applicable to the prediction of the value of the heating and cooling load. Because, the four models show accuracies greater than 90%, the Random forest model being the most suitable for this study as it gives an accuracy of 98%. It is recommended to carry out other ML models with the purpose to improve the prediction results.

Keywords: <ARTIFICIAL INTELLIGENCE>, <ENERGY EFFICIENCY>, <MACHINE LEARNING>, <REGRESSION>, <RESIDENTIAL BUILDINGS>, <ENERGY EFFICIENCY>.



Sandra Paulina Porras Pumalema
C.I. 0603357062

INTRODUCCIÓN

En la actualidad el calentamiento global se ha convertido en el principal problema de la humanidad, puesto que los cambios de temperatura que afectan al planeta, han provocado que las personas se ayuden de diversos sistemas electrónicos para controlar la calefacción y refrigeración dentro de sus hogares, creando así un consumo excesivo de energía, puesto que los edificios residenciales no cuentan con entornos adecuados para el uso de estos aparatos, es así que el rendimiento energético es un parámetro importante a tomar en cuenta al momento de construir un edificio. Dependiendo las condiciones climáticas del entorno y las características que se le dé a la construcción, un edificio puede guardar alrededor del 40% del uso total de la energía. por lo que predecir las cargas de calefacción y refrigeración de un edificio es muy importante, debido a esto se puede decir que la mejor manera para incrementar la eficiencia energética es encontrar soluciones óptimas entre los diferentes diseños en la fase inicial, así como en la fase de funcionamiento después de que el edificio haya sido terminado.

De igual manera se puede decir que para garantizar un consumo de energía sostenible se debe realizar un examen adecuado del rendimiento energético de los edificios (REE) y un diseño óptimo del sistema de calefacción, ventilación y aire acondicionado (CVAC). Aunque en muchos lugares alrededor del mundo han controlado estas medidas, todavía existe un alto consumo de energía y se prevé que aumente aún más. De acuerdo con los elementos mencionados anteriormente, en este estudio se pretende aplicar técnicas de aprendizaje automático basado en regresión lineal, analizando datos reales de edificios residenciales, extraídos de una base de datos del Machine Learning Repository de la Universidad de California, Irvine, los cuales permiten obtener predicciones más cercanas del consumo de energía en diversos entornos, desarrollando así una herramienta de evaluación que ayude a optimizar la eficiencia energética.

Justificación.

La falta de métodos modernos para el análisis de la eficiencia energética y la detección de fallas del consumo energético en edificios residenciales, siempre ha representado un problema, ya que los métodos tradicionales no permiten un correcto monitoreo, ni un análisis adecuado para el control y corrección de estas fallas, generando un consumo excesivo de energía por parte de las personas, mediante diversos estudios los investigadores han demostrado que la aplicación de métodos basados en inteligencia artificial pueden detectar diversos tipos de fallos en relación a consumos y desperdicios de la energía.

La presente investigación se enfocara en demostrar el consumo energético que existe en diversos edificios residenciales a través del análisis de datos de eficiencia energética que serán extraídos del Machine Learning Repository de la Universidad de California, Irvine, el objetivo primordial del estudio es obtener una herramienta de análisis de datos que muestre resultados eficientes y que mejoren el uso de tecnologías de inteligencia artificial aplicadas en el aprendizaje de máquinas y basadas en técnicas de regresión lineal que facilitaran que personal de mantenimiento logre detectar los consumos energéticos, a la vez que logren corregir y mejorar la eficiencia energética dentro de los edificios.

Problema.

El consumo excesivo e innecesario de energía a nivel mundial acompañado de la creación de diversas edificaciones residenciales que no satisfacen las necesidades primordiales del ser humano, así como un irreal análisis de la eficiencia energética y de inexistentes métodos relacionados con el mantenimiento para la detección de fallos del consumo de energía en edificios residenciales, ha provocado que los habitantes de estas infraestructuras, ocasionen grandes pérdidas económicas a los diversos países, además de esto, por las diversas acciones que conllevan el uso indiscriminado de recursos han provocado, grandes daños al medioambiente y etapas de variación de las temperaturas asociadas con el calentamiento global, obligado que la sociedad incremente el uso de dispositivos tecnológico para equiparar estos fenómenos, provocando un aumentado en el consumo de energías y una inestabilidad ecológica para el planeta, por lo que existe la necesidad del uso de la inteligencia artificial para diagnosticar fallos de consumos energéticos en los diversos sectores productivos y residenciales alrededor del mundo, logrando un control adecuado en temas del aprovechamiento de recursos y eficiencia energética.

Hipótesis

Utilizando aprendizaje de máquinas basado en regresión lineal se detecta el consumo de eficiencia energética en edificios residenciales.

Variable dependiente.

Eficiencia energética en edificios residenciales.

Variable independiente.

Precisión.

Error Absoluto medio.

Error Absoluto Cuadrado.

R2.

Objetivos.

Objetivo general.

Analizar la eficiencia energética en edificios residenciales utilizando aprendizaje de máquinas, basados en regresión lineal.

Objetivos específicos.

Realizar el preprocesamiento de los datos de consumo de energía descargados del repositorio de la Universidad de California, Irvine (UCI).

Dividir los datos del repositorio para el entrenamiento y prueba de consumo de energía en edificios residenciales.

Encontrar las herramientas de extracción de datos que mejor correlacionen los datos de consumo de energía.

Diseñar un algoritmo basado en regresión lineal capaz de asociar los datos de consumo de la eficiencia energética en edificios residenciales.

CAPÍTULO I

1. MARCO TEÓRICO CONCEPTUAL

1.1. La eficiencia energética en relación con los edificios residenciales.

Se puede definir la eficiencia energética en edificios residenciales como la utilización racional de energía para abastecer las necesidades energéticas de climatización dentro de un inmueble (calefacción y refrigeración), En otras palabras, la eficiencia energética consiste en la optimización de los recursos utilizados por los equipos para conseguir el mismo rendimiento o realizar la misma función con el fin de obtener un consumo energético eficiente donde se reduzcan las pérdidas de energía, ayude a la disminución de la contaminación ambiental, en otros beneficios como los que se muestran en la figura 1-1.



Gráfico 1-1: Beneficios de la eficiencia energética.

Fuente: (Soluciones, 2021)

Realizado por: Charanchi J. 2022

1.1.1. Variables para la medición de la eficiencia energética en edificios residenciales

A medida que aumenta la densidad poblacional en el planeta, las ciudades tienden a tener un nivel de emisiones de gases de efecto invernadero muy elevado por lo que es de suma importancia calcular el rendimiento energético de los edificios (REE) Uno de los principales factores para medir el REE es la carga de calefacción (CC) y la carga de refrigeración (CR) de una estructura.

Por lo que la predicción de las CC y CR dependen de varias variables, como se explica en la Tabla 1-1. (Sadeghi et al., 2020, p. 1)

Tabla 1-1: Variables necesarias para la medición de la eficiencia energética.

Variables necesarias para la medición de la eficiencia energética					
Variables	Información de las variables	Atributos	Valores Totales	Tipo de datos	Unidades
Variables de entrada	Capacidad relativa	X1	12	Verdadero	Ninguna
	Superficie	X2	12	Verdadero	m^2
	Área de la pared	X3	07	Verdadero	m^2
	Área del techo	X4	04	Verdadero	m^2
	Altura total	X5	02	Verdadero	m^2
	Orientación	X6	04	Entero	Ninguna
	Área de acristalamiento	X7	04	Verdadero	Ninguna
	Distribución del área de acristalamiento	X8	6	Entero	Ninguna
Variables de salida	Carga de calefacción	Y1	586	Verdadero	kWh/m^2
	Carga de refrigeración	Y2	636	Verdadero	kWh/m^2

Fuente: (Sajjad et al., 2020, p. 12)

Realizado por: Charanchi, Javier, 2022

1.1.2. *La eficiencia energética asociada al confort térmico.*

En muchos de los estudios relacionados con los edificios energéticamente eficientes. Todos los modelos de ML desarrollados tienen el mismo objetivo, que es poder predecir las necesidades energéticas futuras y ahorrar energía en los edificios, para tener garantía de esto se lo debe hacer desde el diseño del edificio, lo que en consecuencia conducirá a edificios residenciales más eficientes energéticamente. Las estadísticas de los informes de la Agencia Internacional de Energía (AIE) de 2016 muestran que los edificios son los responsables del 40 % del consumo de toda la energía a nivel mundial y, en consecuencia, contribuyen a casi el 30 % de las emisiones de carbono. (Sekhar Roy, Roy y Balas, 2018, p. 1)

Una vez analizado las estadísticas, se puede decir que prevé que la demanda anual de energía que consumen los edificios aumente una media del 1,5 % entre 2012 y 2040, lo que supondrá un aumento total del 48 % durante ese período. Si no se toman medidas para mejorar la eficiencia energética de los edificios, se espera que la demanda de energía aumente en un 50 % para 2050. (Sajjad et al., 2020, p. 1).

Por lo que se puede determinar que a futuro el principio básico a seguir en las construcciones será la obtención de un equilibrio entre confort térmico y eficiencia energética debido a esto los

ingenieros y arquitectos tienen como prioridad resolver este reto ya que muchos estudios determinan que el confort térmico se encuentra directamente relacionado con la eficiencia energética ya que se busca conseguir un ambiente térmico confortable consumiendo únicamente la energía necesaria teniendo en cuenta que el aspecto importante que se debe tomar en cuenta es que la eficiencia energética nunca, debe comprometer el confort interior para los usuarios de los edificios.(Tristancho Carvajal, 2019, p. 15).

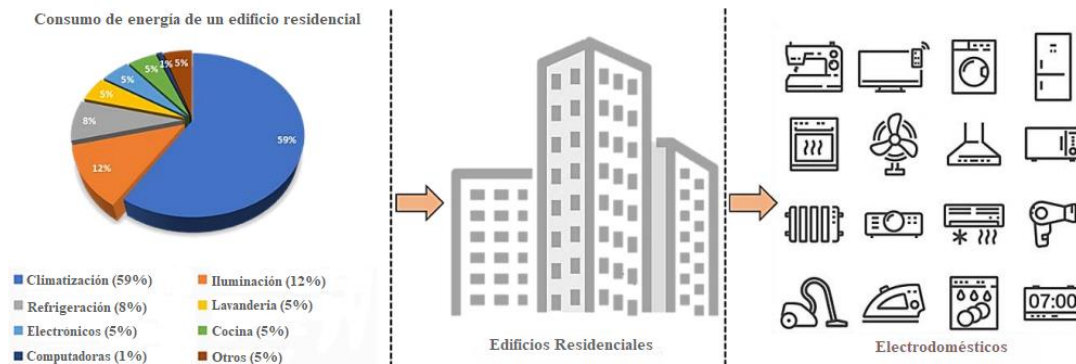


Gráfico 2-1: Consumo de energía de un edificio residencial

Fuente: (Sajjad et al., 2020)

Realizado por: Charanchi J. 2022

1.1.3. *Impulsores de valor basados en la eficiencia energética.*

El diseño de edificios energéticamente eficientes acompañado de métodos de I.A destinados a el análisis de consumo energético, ha permitido que los ingenieros y arquitectos aborden objetivos que van más allá de la mejora del medio ambiente. En este sentido, diversos estudios han determinado factores de valor que se persiguen con los edificios sostenibles como lo son las mejoras ambientales, sociales, económicas, sanitarias y de confort térmico. La mejora medioambiental es evidente. Si se reduce el consumo de energía del edificio, se espera que las emisiones de gases de efecto invernadero sean menores. Asimismo, el uso de materiales sostenibles podría garantizar que el impacto ambiental durante la construcción sea bajo. Esta mejora ambiental se produce junto con una mejora del confort y la salud de los usuarios.(Bienvenido-Huertas y Rubio-Bellido, 2021, pp. 4).

Los proyectos de eficiencia energética en edificios deben entenderse no solo como una mejora de las condiciones de habitabilidad de los usuarios o la reducción de los impactos ambientales de los edificios, sino también como un impacto directo en la valoración económica de los mismos(Warren-Myers, Kain y Davidson, 2020, pp. 1). Estudios recientes de la Royal Institution of Chartered Surveyors han subrayado que la eficiencia energética de los edificios se está

convirtiéndose en un impulsor de valor en la financiación de proyectos de construcción por parte del sector bancario, Además, una mejor etiqueta energética de los edificios contribuye a un mayor valor monetario de las viviendas. (Bienvenido-Huertas y Rubio-Bellido, 2021, pp. 5).

Sin embargo, existe un problema en el argumento del generador de valor de la eficiencia energética de los edificios en términos de valoración de la propiedad. Los generadores de valor tradicionales se relacionan con los aspectos visibles de los edificios (por ejemplo, el acristalamiento adecuado o el estado de las instalaciones) y son más importantes en la evaluación de los edificios que los datos energéticos, como la certificación energética. En el futuro, la eficiencia energética será probablemente más importante en la relación existente entre prestamistas e inversores y contribuirá a varios préstamos a plazos (RICS, 2019, pp. 5).

Finalmente, es importante enfatizar que la generación de valor en proyectos de eficiencia energética de edificios para reducir la Pobreza Energética (PE). Un hogar está en PE cuando más del 10% de sus ingresos se destina a pagar la factura de la luz. La posibilidad de PE en un hogar depende por tanto de la renta familiar o de su gasto energético (Attia et al., 2017, p. 242). El nivel de ingresos de los usuarios y sus posibles variaciones que se podrían producir en las familias (por ej, por la pérdida de empleo de sus miembros) podrían ser desconocidos al momento de diseñar un nuevo edificio, por lo que un generador de valor para los proyectos de eficiencia energética es que estos proyectos constituyen instrumentos para reducir la posibilidad de casos de PE en el futuro. La adquisición de una unidad de vivienda con mejor desempeño energético asegura que es poco probable que una familia esté en EP en el futuro debido al desempeño energético de la unidad de vivienda. (Bienvenido-Huertas y Rubio-Bellido, 2021, p. 5).

1.2. Introducción a Anaconda con Python.

En los últimos años con el avance de la tecnología hacia la industria 4.0, la programación se ha convertido en un pilar fundamental para el desarrollo de la inteligencia artificial y el aprendizaje de máquinas, convirtiendo a el lenguaje de programación Python con su distribución de código abierto Anaconda en uno de los más populares debido a su naturaleza de ser un software gratuito de código abierto, así como la gran acogida de los usuarios en línea.

1.2.1. Anaconda.

Anaconda es un software gratuito que le proporciona un conjunto de herramientas diseñado para la investigación y la ciencia. La instalación de Anaconda le brinda acceso a diferentes entornos que le permiten codificar en Python o R. Estos entornos, también conocidos como entornos de

desarrollo integrados (IDE), son plataformas o aplicaciones que facilitan enormemente el desarrollo de código. Tienen una función similar a los procesadores de texto como Microsoft Word, Google Doc. y Páginas para escribir texto y mucho más. (Rolon-Mérette et al., 2020, p. 2).

1.2.2. *Python.*

Python es un lenguaje de programación de alto nivel, interpretado, orientado a objetos y con semántica dinámica. Su sintaxis es simple y fácil de aprender, lo que enfatiza la legibilidad, por tanto, reduce el coste de mantenimiento del programa. Soporta módulos y paquetes, con lo que se fomenta la modularidad del programa y la reutilización del código. Es un lenguaje muy atractivo debido al aumento de productividad que proporciona gracias a la carencia de un paso de compilación, hecho que implica que el ciclo de edición-prueba-depuración sea muy rápido. El intérprete de Python y la extensa biblioteca estándar están disponibles en forma de código fuente o binaria sin cargo para todas las plataformas principales, y pueden ser distribuidos libremente. (Python Software Foundation, 2021).

(Arif, 2020, p. 1), en el libro “Introduction to Deep Learning for Engineers” dice que Python es un lenguaje de programación popular orientado a objetos, que fue creado inicialmente por Guido van Rossum en 1990. Más tarde, muchos desarrolladores y programadores contribuyeron a su crecimiento, y su popularidad que ha aumentado en los últimos años en muchos dominios diferentes, incluido el aprendizaje automático.

1.2.2.1. *¿Por qué aprender Python?*

El lenguaje de programación Python se ha convertido en el lenguaje predilecto para muchas aplicaciones de ciencia de datos debido a que combina el poder de los lenguajes de programación de propósito general con la facilidad de uso de lenguajes de secuencia de comandos específicos como MATLAB o R, además de esto Python cuenta con diversas bibliotecas destinadas a los datos de carga, visualización, estadísticas, procesamiento de lenguaje natural, procesamiento de imágenes y más. Esta variedad de herramientas proporciona a los científicos una gama amplia de datos con funcionalidad y propósito especial. Una de las principales ventajas de usar Python es la capacidad de interactuar directamente con el código, utilizando una terminal u otras herramientas como Jupyter Notebook. El aprendizaje automático y el análisis de datos son procesos mentalmente interactivos, en los que los datos impulsan el análisis por lo que es fundamental para estos procesos, tener herramientas que permitan una iteración rápida y fácil, como un lenguaje de programación de propósito general, Python también permite la creación de complejas interfaces

gráficas de usuario (GUI), servicios web, y algoritmos para la integración de sistemas existentes. (Müller y Guido, 2017, p. 5).

(Rolon-Mérette et al., 2020) dice que Python es un lenguaje de programación de nivel medio, interpretado y orientado a objetos lo que lo ase fácil de aprender y usar, al mismo tiempo, que es lo suficientemente versátil para abordar una variedad de tareas, al ser un software libre y tener una compatibilidad con todos los sistemas operativos que existen hoy en día entre los cuales están (Mac, Windows, Linux, Ubuntu), además de tener bajos requerimientos de operación del sistema, hace que cualquier persona pueda codificar con Python. cabe mencionar que desde el individuo más común hasta los mejores científicos de investigación pueden y han desarrollado múltiples proyectos relacionados con el aprendizaje automático, la inteligencia artificial y más. Proyectos que son fácil de encontrar, ya que solo con poner la palabra clave “Python” a cualquier consulta, nos proporciona una gran cantidad de códigos fuentes, cursos, recursos y soluciones en manera de tutoriales para los diversos problemas que se nos pueda presentar, etc.

Finalmente se puede concluir que las diversas ventajas que presenta el lenguaje de programación Python con su distribución de código abierto Anaconda lo convierten en el software ideal para el análisis y captura de datos, el entrenamiento y la creación de algoritmos para la detección de fallos, creando así una herramienta práctica para los diversos procesos de mantenimiento preventivo.

1.3. Inteligencia artificial.

La inteligencia artificial o IA es la capacidad de las máquinas para usar algoritmos, aprender de los datos y utilizar lo aprendido en la toma de decisiones tal y como lo haría un ser humano Sin embargo, la diferencia de las personas y los dispositivos basados en IA es que no necesitan descansar y pueden analizar grandes volúmenes de información a la vez. Asimismo, la proporción de errores es significativamente menor en las máquinas que realizan las mismas tareas que sus contrapartes humanas. (Rouhiainen, 2018, p. 17).

1.3.1. *Importancia de la inteligencia artificial.*

Con la idea de que los equipos y las máquinas puedan por si solos aprender a tomar decisiones, los investigadores han determinado que es de suma importancia el aumento de métodos y herramientas basadas en ML, debido a que sus procesos evolucionan exponencialmente con el tiempo. Gracias a estas dos capacidades, los sistemas de inteligencia artificial ahora pueden realizar muchas tareas que antes solo estaban disponibles para los humanos. Las tecnologías

basadas en inteligencia artificial se han utilizado para ayudar a los humanos a beneficiarse de mejoras significativas y una mayor eficiencia. en casi todos los hábitos de la vida diaria.(Rouhiainen, 2018, p. 17).

1.3.2. Aplicaciones más comunes de la inteligencia artificial.

La IA es una de las herramientas más utilizadas hoy en día y una de las cuales se puede aplicar en la mayoría de las situaciones y actividades. Estas son algunas de las aplicaciones técnicas de la IA con el mayor crecimiento en la actualidad:

- Reconocimiento de imágenes estáticas, clasificación y etiquetado.
- Mejoras del desempeño de la estrategia algorítmica comercial.
- Procesamiento eficiente y escalable de datos de pacientes.
- Mantenimiento predictivo.
- Detección y clasificación de objetos.
- Distribución de contenido en las redes sociales.
- Protección contra amenazas de seguridad cibernética (Rouhiainen, 2018, p. 18).

1.4. Machine learning (ML).

El aprendizaje automático (en inglés, *machine learning*) es uno de los enfoques principales de la inteligencia artificial. En pocas palabras, se trata un aspecto de la informática en el que los ordenadores o las máquinas tienen la capacidad de aprender sin estar programados para ello.(Rouhiainen, 2018, p. 19) Así también (Zhang, 2017, p. 223) dice que el término Machine learning (ML) es un subconjunto de la inteligencia artificial, que construye un modelo matemático basado en datos de muestra, conocido como "datos de entrenamiento", con el fin de hacer predicciones o decisiones sin estar programado explícitamente para realizar la tarea.

ML resuelve situaciones por sí solo a partir de un análisis de datos y cuantos más datos tengan mejores resultados, además, para realizar el análisis se utilizan algoritmos que diseñan otros datos según las necesidades. A través de los datos de entrada, ML ejecuta un algoritmo y como resultado, genera más información para el problema.(Manrique, 2020, p. 587) Sin embargo una de las principales dificultades a la hora de aplicar un proceso de ML a un conjunto de datos preestablecido es la elección del flujo de trabajo adecuado, ya que en la teoría existen muchos enfoques diferentes a un mismo problema, por lo que dependiendo del origen de los datos y los objetivos del análisis. Se debería escoger el algoritmo (Cardoso y Ferreira, 2021, p. 2).

Por lo que se podría decir los algoritmos son un conjunto de instrucciones explícitamente programadas que utilizan las computadoras para calcular o resolver un problema. Es decir, los algoritmos de ML permiten que las computadoras construyan modelos a partir de datos de muestra disponibles y automaticen el proceso de toma de decisiones, basándose en la entrada de datos y la experiencia. Estas técnicas identifican patrones en los datos y proporcionan varias herramientas para la minería de datos (Shobha y Rangaswamy, 2018, p. 1).

1.4.1. Tipos de aprendizaje en machine learning.

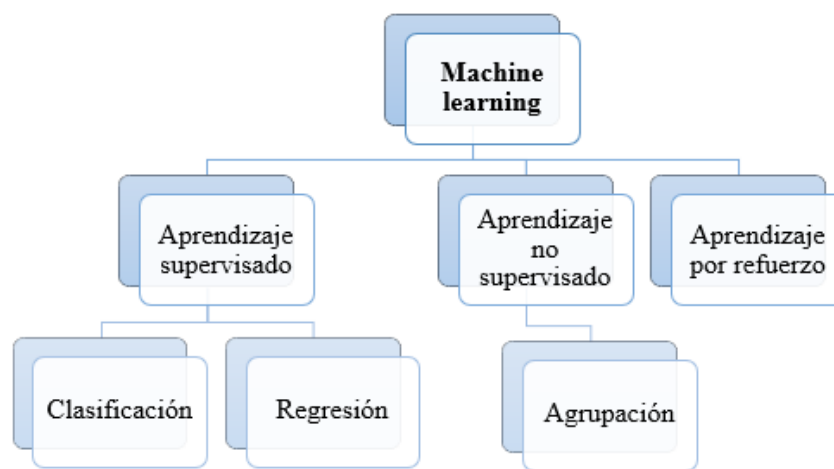


Gráfico 3-1: Tipos de machine learning

Fuente:(Shobha y Rangaswamy, 2018, p. 199)

Realizado por: Charanchi J. 2022

1.4.1.1. Aprendizaje supervisado.

Los enfoques del aprendizaje supervisado se crean algorítmicamente a partir de un conjunto de datos, dando una función que toma como entrada un vector y luego emite un valor predicho para cada punto de datos. Más formalmente, estos métodos aprenden de una función, F , que predice una variable de respuesta, y , desde un vector de características, X , contenido en N variables de entrada, tales que $f(x) = y$. Si y es una variable categórica, nos referimos a la tarea como una clasificación de problema, mientras que si y es una variable continua, nos referimos como regresión. En el aprendizaje supervisado, el objetivo es optimizar $f: x \rightarrow y$ utilizando un 'conjunto de entrenamiento' de datos etiquetados (es decir, cuyos valores de respuesta se conocen). A la vez que, asumimos que tenemos un conjunto de datos de entrenamiento de longitud n de la forma $\{(X_1, Y_1), \dots, (X_n, Y_n)\}$, donde $X \in R^M$. Existe una variedad de algoritmos de aprendizaje que pueden crear funciones que pueden realizar clasificación o regresión, incluido support vector machines (SVM), árboles de decisión, random forests, boosting y redes neuronales artificiales

(ANN) que en su forma moderna están asumidos bajo el paraguas del aprendizaje profundo. Estos algoritmos difieren se estructuran y entrenan f (Schrider y Kern, 2018, p. 4).

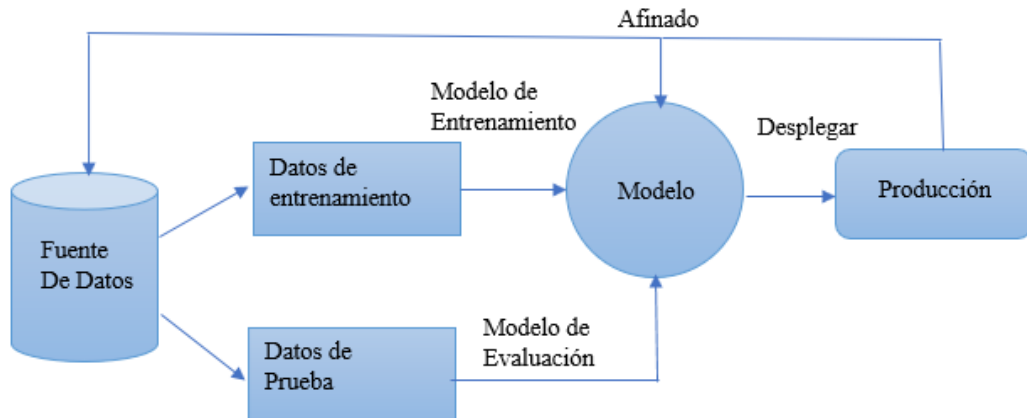


Gráfico 4-1: Flujo de trabajo del aprendizaje supervisado

Fuente: (Batta, 2020, p. 381)

Realizado por: Charanchi J. 2022

1.4.1.2. *Aprendizaje no supervisado.*

Los algoritmos de aprendizaje no supervisados generalmente se aplican a datos que no contienen información de etiqueta, por lo que no sabemos cuál es el derecho, solo la salida. Por lo tanto, es muy difícil decir si un modelo "funcionó bien". Para ejemplo, nuestro hipotético algoritmo de agrupamiento podría haber agrupado todas las imágenes que muestran caras de perfil y todas las imágenes de caras completas. Esto ciertamente puede ser una forma posible de dividir una colección de imágenes de rostros de personas, pero no es la única que estábamos buscando. Sin embargo, no hay forma de que podamos "decirle" al algoritmo lo que están buscando y, a menudo, la única forma de evaluar el resultado de un algoritmo no supervisado es inspeccionarlo manualmente.

Como consecuencia, los algoritmos no supervisados se utilizan a menudo en un entorno exploratorio, cuando un científico de datos quiere comprender mejor los datos, en lugar de como parte de un sistema automático más grande. Otra aplicación común para algoritmos no supervisados es como un paso de preprocesamiento para algoritmos supervisados. Aprendiendo una nueva representación de los datos a veces pueden mejorar la precisión de los algoritmos supervisados, o pueden conducir a la reducción del consumo de memoria y tiempo.(Müller y Guido, 2017, p. 132).

En otras palabras, el aprendizaje no supervisado tiene como objetivo encontrar las regularidades en la entrada de la función, de modo que ciertos patrones ocurran con más frecuencia que otros y aprender a ver qué sucede generalmente y qué no. Los ejemplos más comunes son: El reconocimiento de voz, la agrupación de documentos y la compresión de imágenes.(Shobha y Rangaswamy, 2018, p. 200).

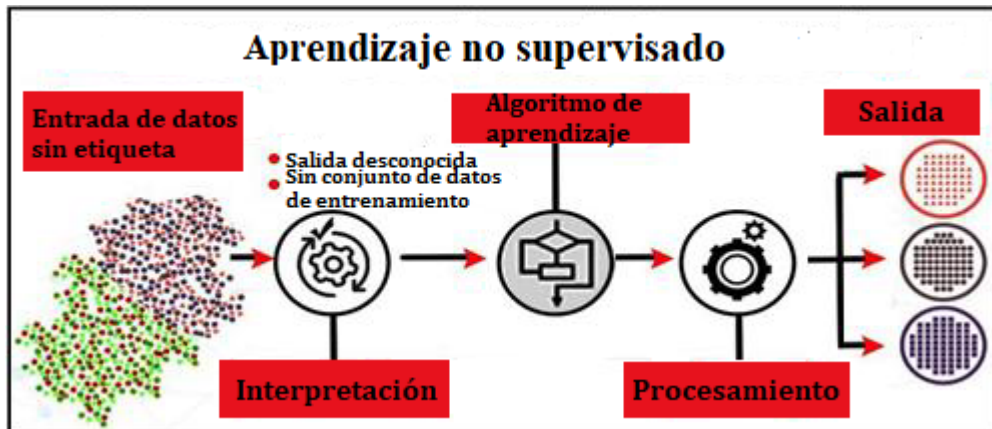


Gráfico 5-1: Flujo de trabajo del aprendizaje no supervisado

Fuente: (Batta, 2020, p. 383)

Realizado por: Charanchi J. 2022

1.4.1.3. *Aprendizaje por refuerzo.*

El aprendizaje por refuerzo es un área del aprendizaje automático que se ocupa de cómo los agentes de software deben realizar acciones en un entorno para maximizar alguna noción de recompensa acumulativa. El aprendizaje por refuerzo aprende de es uno de los tres paradigmas básicos del aprendizaje automático, junto con el aprendizaje supervisado y el aprendizaje no supervisado.

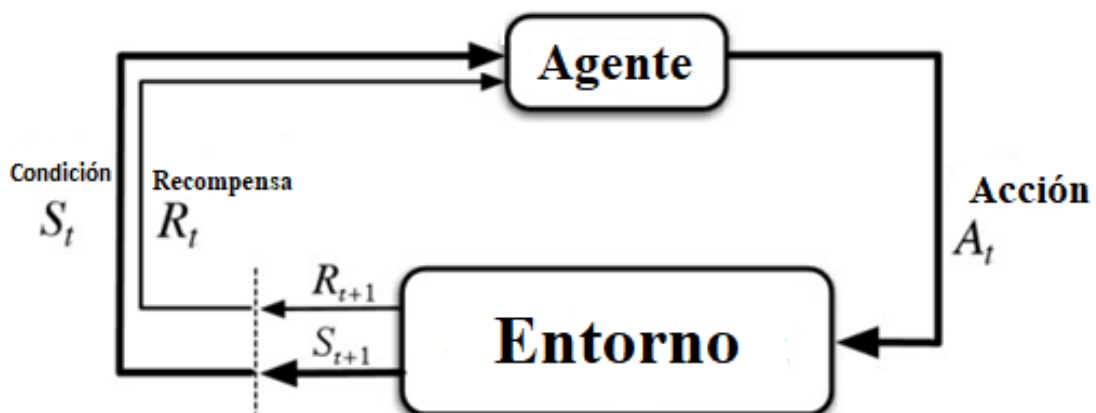


Gráfico 6-1: Flujo de trabajo del aprendizaje por refuerzo

Fuente: (Batta, 2020, p. 384)

Realizado por: Charanchi J. 2022

1.4.2. Pasos para construir un modelo de machine learning.

La creación de un modelo de ML no es un proceso que se limite a uso de un algoritmo de aprendizaje o una biblioteca de ML; es un proceso que generalmente implica al menos 6 pasos, los cuales se muestran en la Figura 7-1. (Manrique, 2020, p. 589).

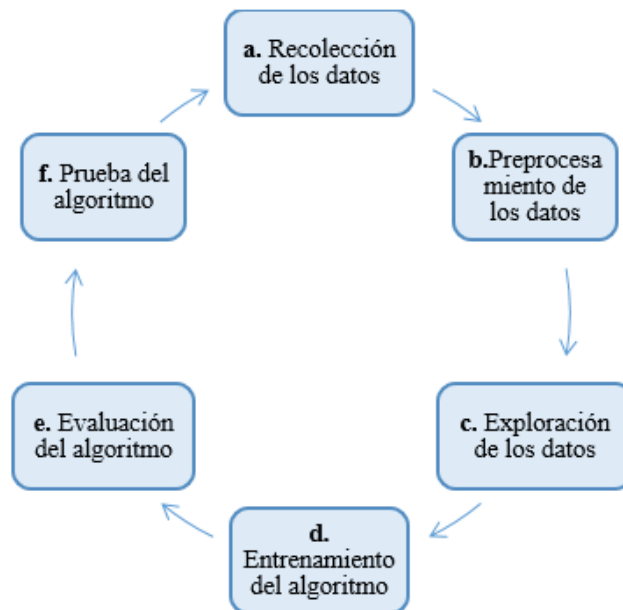


Gráfico 7-1: Proceso de construcción del modelo de aprendizaje automático

Fuente: (Manrique, 2020, p. 589)

Realizado por: Charanchi J. 2022

(Manrique, 2020, p. 289), describe cada uno de estos pasos como se describe a continuación:

a. Recolección de datos: Los datos para la creación del algoritmo de aprendizaje se lo puede realizar a través de bases de datos en repositos.

b. Preprocesamiento de los datos: Con los datos disponibles, se debe asegurar que todos tengan un formato correcto para alimentar el algoritmo de aprendizaje. Por lo general se tiene que realizar varias tareas de preprocesamiento antes de poder usar los datos.

c. Exploración de los datos: Se realiza un análisis previo para corregir los casos de valores faltantes o tratar de encontrar a primera vista cualquier patrón en ellos que facilite la construcción del modelo. En este punto, se deben detectar valores atípicos; o encuentre las características que tienen más influencia para hacer una predicción.

d. Entrenamiento del algoritmo: los algoritmos de aprendizaje se alimentan con los datos que se procesaron en las etapas anteriores con el propósito que los algoritmos puedan extraer la información relevante del conjunto de datos y luego hacer las predicciones.

e. Evaluación del algoritmo: Se realizan las pruebas de la información que genera el conocimiento del entrenamiento previo que se obtuvo a través del algoritmo.

f. Prueba del algoritmo: Se realiza una evaluación sobre la precisión del algoritmo en sus predicciones y, si no está satisfecho con su rendimiento, debe volver a la etapa anterior y continuar entrenando el algoritmo cambiando algunos parámetros hasta que se logre un rendimiento aceptable.

1.4.3. Problemas del machine learning más comunes.

El aprendizaje automático es una de las herramientas tecnológicas que está estrechamente relacionado con la inteligencia artificial y el aprendizaje estadístico. Debido a esto, incorpora algoritmos de procedimientos desarrollados los cuales son utilizados para describir los mismos algoritmos, procedimientos y parámetros del modelo. (Villena et al., 2019, p. 4).

Como principales problemas de aprendizaje automático se incluyen los siguientes:

- **Clasificación o reconocimiento de patrones:** el problema de enseñar a un clasificador a asignar etiquetas a nuevas muestras sin etiqueta.
- **Regresión:** el problema de estimar la relación entre salidas y atributos de valor real para hacer predicciones.
- **Agrupación en clúster:** la tarea de agrupar datos de manera que los ejemplos en el mismo grupo (llamado agrupación) sean más similares entre sí que a los de otros grupos.
- **Recomendación:** la tarea de priorizar ejemplos basados en los atributos de interés.
- **Reducción de la dimensionalidad:** el problema de transformar atributos en un espacio de alta dimensión a un espacio de menos dimensiones.
- **Análisis de redes:** el estudio de explorar asociaciones entre componentes de sistemas para comprender la función biológica de componentes individuales y dilucidar los comportamientos de los sistemas biológicos.
- **Estimación de densidad:** el problema de estimar la función de densidad de probabilidad para una población en base a los datos observados.

Como (McArthur et al., 2018) Indican, un desafío clave es la calidad de los datos. Por lo general, se recopilan datos que con frecuencia carecen de los detalles necesarios para el diagnóstico del problema.

1.5. Principales modelos de machine learning basados en de regresión.

Entre las múltiples técnicas utilizadas en ML para el pronóstico y predicción de las cargas de calefacción (C.C) y las cargas de refrigeración (C.R) en edificios residenciales, están los modelos de regresiones, los cuales son algoritmos de regresión que se ocupan de modelar la relación entre variables que se refinan iterativamente utilizando una medida de error en las predicciones realizadas por el modelo lo que permiten describir como influye una variable x sobre una variable y, estos modelos pueden ser de carácter simple o múltiple dependiendo el caso como se puede observar en la figura 8-1, (Shobha y Rangaswamy, 2018, p. 11).

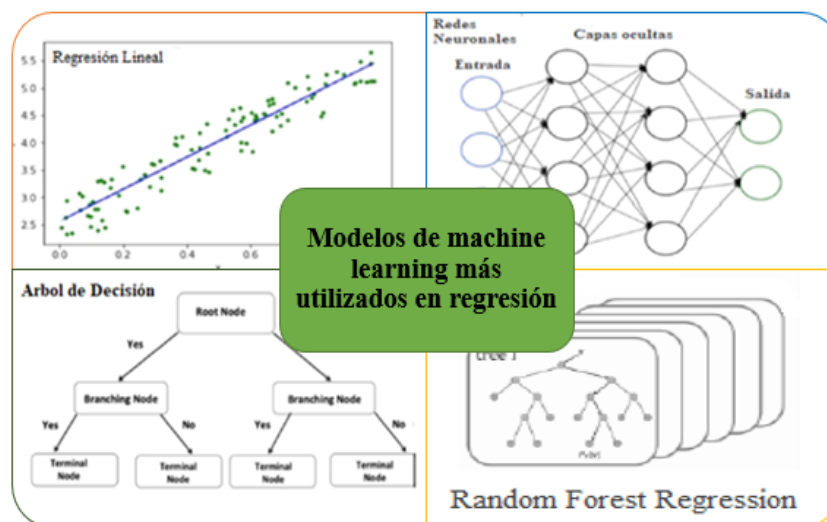


Gráfico 8-1: Modelos de machine learning más utilizados en regresión.

Fuente: (Lin et al., 2020, p. 8)

Realizado por: Charanchi J. 2022

1.5.1. Regresión lineal (RL).

La regresión lineal (RL) es uno de los métodos estadísticos más simples que modela la relación entre variables continuas ajustando los datos observados a una ecuación lineal. Para ajustar el modelo, primero es necesario verificar si existe alguna relación entre las variables de interés, lo cual se hace utilizando una variable numérica llamada coeficiente de correlación.

Una línea de regresión lineal se representa con la ecuación de la forma:

$$Y = a + bx \quad (1)$$

Donde x es la variable independiente, y es la variable dependiente, b es la línea de la pendiente y a es la intersección (donde el valor de y es dado cuando $x = 0$) como se ve en la figura 9-1. Para encontrar la línea mejor ajustada, una de las medidas más comunes que se utilizan son los errores de mínimos cuadrados, que se hace con el objetivo de minimizar la suma de los cuadrados de la desviación vertical de cada punto de la línea o la suma de los cuadrados de los derechos residuales. (Sekhar Roy, Roy y Balas, 2018, p. 4).

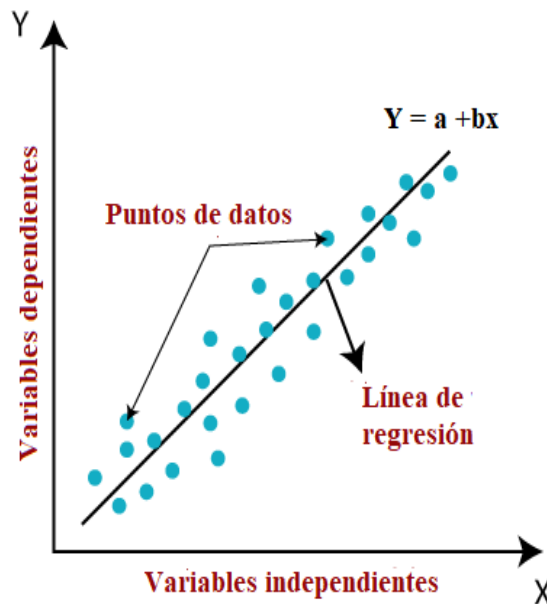


Gráfico 9-1: Representación de un proceso de regresión lineal.

Fuente: (Quora, 2017)

Realizado por: Charanchi J. 2022

1.5.1.1. *Importancia de la regresión lineal.*

La regresión lineal (RL) es probablemente la técnica estadística más simple y la más común para medir las relaciones entre variables continuas, reales o numéricas, por lo que lo convierte en el método ideal para las predicciones en las investigaciones de carácter cuantitativo. Dicha técnica se encuentra consolidada y posee muchos años de existencia debido a ello existe una crítica de que su efectividad, por lo que algunos investigadores no piensan que sea la más adecuada para la obtención de resultados debido a su antigüedad y obsolescencia. Sin embargo, existe una

contraparte que afirma que la RL funciona efectivamente y al ser una técnica con tanto tiempo de uso se debe a los resultados precisos que esta puede dar. (Mejía, 2019, p. 43).

Según (Hope, 2019, p. 1) nos dice que el análisis de regresión lineal es probablemente la forma más sencilla y popular de medir las relaciones entre el predictor continuo y las variables de respuesta. El método asume que las relaciones entre el predictor y las variables objetivas son lineales: es decir, una unidad de cambio constante en una implica una unidad de cambio constante en la otra. Esta simplicidad generalmente hace que la regresión lineal sea la opción óptima para análisis con muestras pequeñas y también hace que estos modelos sean relativamente fáciles de interpretar y comprender.

1.5.2. *Árbol de decisión (AD).*

Los árboles de decisión (AD) son modelos de ML que construyen modelos de clasificación o regresión en forma de estructura de árbol descomponiendo un conjunto de datos establecido en subconjuntos pequeños mientras que al mismo tiempo una rama del árbol se desarrolla gradualmente. Obteniendo como resultado un árbol con nodos de decisión y hojas. (Duarte et al., 2017, p. 6).

Los AD toman un conjunto de atributos como entrada y devuelven un valor predicho para la entrada respectiva. Se toma una decisión, asociada con el nodo de decisión. ejecutando una secuencia de prueba, cada nodo interno del árbol corresponde a una prueba del valor de las propiedades y las ramas de este nodo identificando posibles valores de prueba. Cada nodo especifica el valor devuelto si se alcanza la hoja. En este método, el parámetro estimado es la profundidad máxima del árbol. (Duarte et al., 2017, p. 6).

1.5.3. *Random Forest Regression (RFR).*

Random Forest, o bosque de decisión aleatorio es un modelo de ML propuesto por Breiman en el año 2001, define a Random Forest como un método de aprendizaje para problemas de clasificación y regresión. (Qiu et al., 2017, p. 8).

Según (Wang et al., 2018, p. 8), RF es un modelo de predicción por conjuntos que consiste en una colección de diferentes árboles de regresión, que son entrenados a través de un embolsado y selección de variables aleatorias.

El modelo RF integra múltiples árboles de decisión para mitigar la varianza del modelo sin aumentar el sesgo. La radiofrecuencia combinada con embolsado y un subconjunto aleatorio de funciones hacen que este modelo se caracteriza por ser rápido y robusto al ruido de los datos por lo que la combinación de árboles en RF puede reducir el error en la resolución de problemas de regresión porque utiliza la agregación bootstrap o harpillera. La principal ventaja de RF es reducir el error de la predicción al considerar todos los árboles de decisión en el bosque y su correlación con las predicciones. (Pham et al., 2020, p. 3).

Según la matemática un modelo de RF es un conjunto de árboles C representado por $T_1(X), T_2(X), \dots, T_C(X)$, donde $X = x_1, x_2, \dots, x_m$ es un vector de entradas de m dimensión. El conjunto resultante produce salidas C como se define a continuación: (Pham et al., 2020, p. 3).

$$Y_{\text{pred}_1} = T_1(X), Y_{\text{pred}_2} = T_1(X), \dots, Y_{\text{pred}_c} = T_c(X) \quad (2)$$

Donde Y_{pred_c} es el valor de predicción obtenido por el árbol de decisión C .

La salida de todos estos árboles generados aleatoriamente se agrega para obtener una predicción final Y_{pred_c} que son los valores promedio de todos los árboles en el bosque. El modelo RF genera un número C de árboles de decisión a partir de N número de muestras de entrenamiento. Para cada árbol en el bosque, arranca con un muestreo donde se realiza la creación de nuevos conjuntos de entrenamiento, mientras que las muestras no seleccionadas se denominan conjuntos listos para usar. Así también usa el muestreo bootstrap que es un método de remuestreo realizado de forma independiente al muestreo con reemplazo de los datos originales con el mismo tamaño de la muestra. Esto hace que el nuevo conjunto de entrenamiento se use luego para crear completamente un árbol de regresión (o clasificación) no podado. Usando el azar y la selección de características, se selecciona un pequeño número de m características (variables de entrada) seleccionado aleatoriamente en cada división de un nodo del árbol de decisión. El proceso se repite hasta que se cultivan M árboles de decisión para formar un bosque generado. (Pham et al., 2020, p. 3).

En la Grafico 10-1 se presenta la estructura de un modelo de RF para predecir el consumo de energía en edificios. Los datos de energía del edificio se dividieron en datos de entrenamiento y datos de prueba. La efectividad del modelo de RF en la predicción se examinó a través de los procesos de capacitación y prueba.

El proceso de formación de un bosque generado aleatoriamente se describe a continuación:

1. Se crea un remuestreo de datos usando el método bootstrap del entrenamiento del conjunto de datos
2. Se hace crecer un árbol para cada muestra de arranque anterior en la que se divide entre un subconjunto seleccionado aleatoriamente de variables de entrada, que es el parámetro de ajuste de la radiofrecuencia.
3. Se repite los pasos 1 y 2 hasta que crezcan los árboles C.

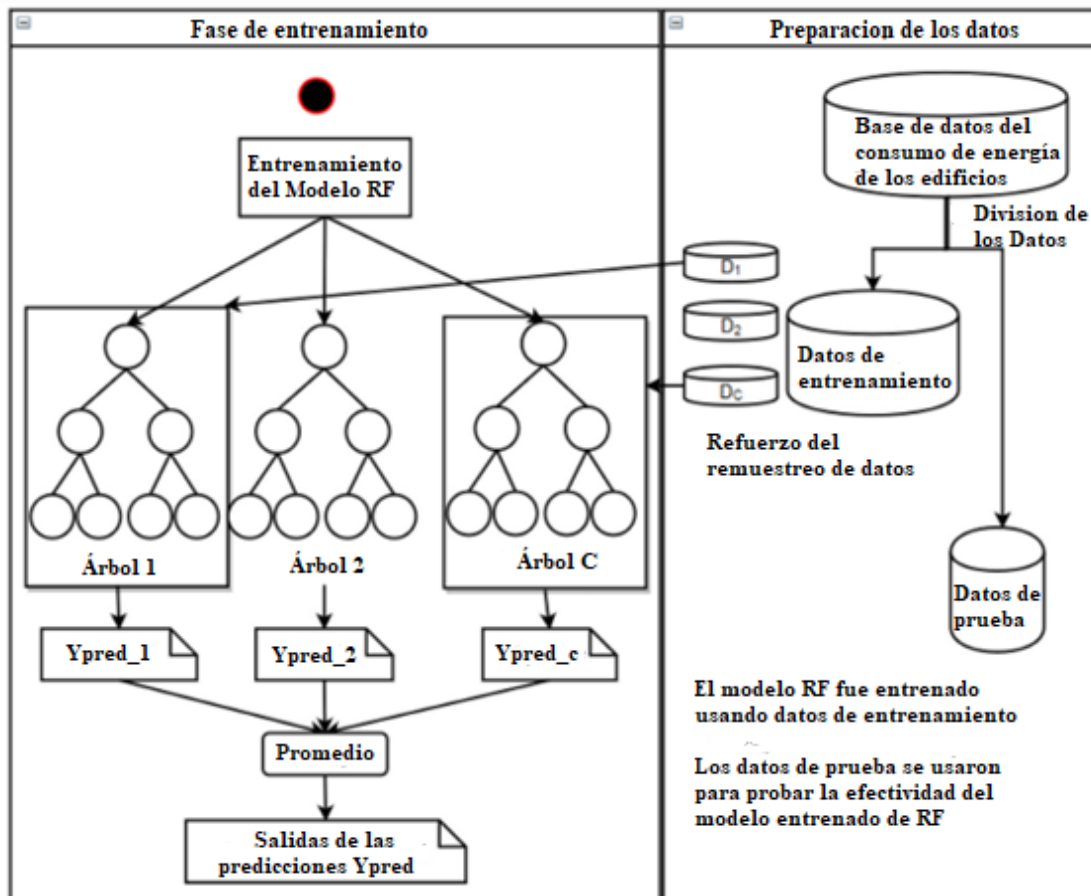


Gráfico 10-1: Proceso de construcción de un modelo de Random Forest

Fuente: (Pham et al., 2020, p. 4)

Realizado por: Charanchi J. 2022

1.6. Métricas de evaluación de un modelo de ML basado en regresión.

Para evaluar un modelo ML de aprendizaje supervisado basado en regresión, donde el modelo aprende a predecir puntuaciones numéricas, es necesario determinar el desempeño del modelo, por lo que es necesario calcular varios rendimientos utilizando diferentes parámetros entre los cuales está, el subconjunto de entrenamiento y el subconjunto de prueba, así también se debe tener en cuenta la confianza estadística. Por lo que se utilizan tres criterios de evaluación de rendimiento en cada uno de los modelos presentados en este trabajo. Estos son medidas de error, es decir, el

error cuadrado (R^2), el error cuadrático medio (RMSE) y el error absoluto medio (MAE), descritos en las Ecuaciones (3), (4) y (5), respectivamente. Estos errores han sido elegidos porque son los utilizados con mayor frecuencia en estudios previos. (Nebot y Mugica, 2020, p. 8).

Tanto RMSE como MAE son medidas de precisión de uso común que son útiles al comparar diferentes métodos sobre los mismos datos, como es el caso de esta investigación. El MAE es una medición de puntuación lineal, lo que significa que todas las medidas y las diferencias individuales se ponderan por igual en un promedio. El RMSE da un peso relativamente alto a los errores grandes, ya que los errores se elevan al cuadrado antes de promediarlos. Esto significa que el RMSE es más útil cuando los errores grandes son particularmente indeseables. El MAE y el RMSE se puede usar en conjunto para diagnosticar la variación en los errores en un conjunto de pronósticos. (Nebot y Mugica, 2020, p. 8).

Los errores se calculan de la siguiente manera:

La Ecuación (3) corresponde al R^2 , que representa la varianza porcentual en la variable dependiente explicada por las independientes. (Sadeghi et al., 2020, p. 11).

$$R^2 = \frac{\sum_{t=1}^n (y_t - x_t)^2}{\sum_{t=1}^n (y_t - \bar{y})^2} \quad (3)$$

Donde n es el tamaño de la muestra, x_t es el valor predicho, y_t es el valor observado y \bar{y} es la media de y_t .

La ecuación (4) correspondiente al RMSE que representa la desviación estándar de la muestra de los residuos entre el valor previsto y los valores observados. Esta medida se utiliza para identificar grandes errores y evaluar la fluctuación de respuesta del modelo con respecto a la varianza. (Wang et al., 2018, p. 12).

$$RMSE = \sqrt{\frac{1}{n} \sum_{t=1}^n (x_t - y_t)^2} \quad (4)$$

La ecuación (5) correspondiente al MAE que representa la magnitud de la diferencia entre el valor de la predicción y el valor real que existe en un modelo.

$$\text{MAE} = \frac{1}{N} \sum |x_t - y_t| \quad (5)$$

1.7. Detección del consumo de energía utilizando aprendizaje de máquinas.

Con el surgimiento de nuevas tecnologías en el campo de la computación para el análisis de datos y la detección de fallas, la necesidad del implemento de herramientas basadas en aprendizaje automático han contribuido recientemente al desarrollo de modelos predictivos en varios campos (Ünlü y Namli, 2020, p. 1). Estos modelos mejoran las capacidades de análisis de datos con una mejora en la apreciación, precisión y visibilidad, además que generalizan el uso de las herramientas de pronóstico para series temporales existentes. Recientemente, los investigadores se han centrado en las aplicaciones de las técnicas de ML para pronosticar la producción de energías renovables, energías no renovables y el consumo de éstas.(Khan et al., 2021, p. 1).

En múltiples países alrededor del mundo existen ministerios y organismos afines al sector de la energía, los cuales están enfocados en controlar el consumo de la energía en general por lo que el análisis de las fuentes de datos disponibles en diversos repositorios y el estudio de diferentes dimensiones tomadas del análisis que surgen de estas investigaciones, como los datos de uso de gas natural, electricidad, las características de los edificios residenciales, el rendimiento energético, la refrigeración y la calefacción, o datos de clima y pronóstico del tiempo, todo esto acompañado de modelos predictivos basados en ML, logran que las autoridades públicas tengan una herramienta técnica para verificar el consumo energético dentro de sus países, también cabe mencionar que los resultados arrojados por estas investigaciones se utilizan para que las autoridades orienten el comportamiento de los ciudadanos hacia usos más eficientes de energía, adoptando enfoques estructurados basados en bases científicas a través de medidas legales, subsidios económicos o iniciativas de difusión de buenas prácticas.(Abdelaziz, Santos y Dias, 2021, p. 2).

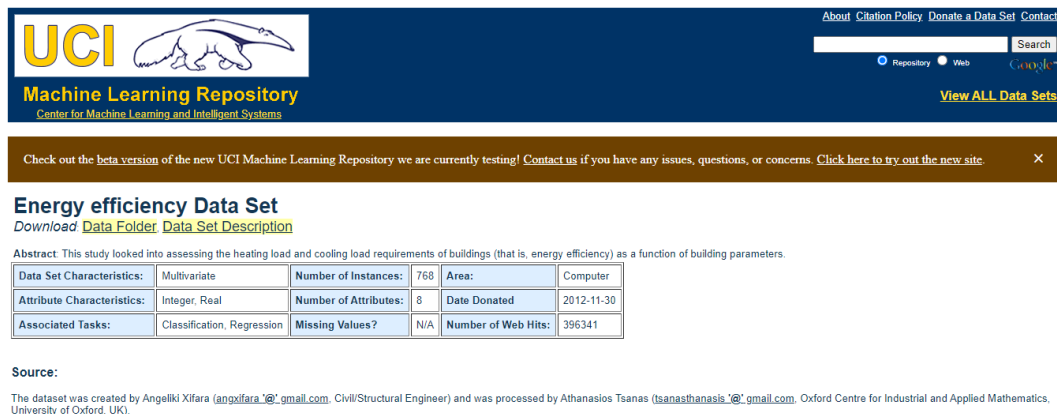
CÁPITULO II

2. MARCO METODOLÓGICO.

El presente capítulo tiene como objetivo principal describir la metodología que se utilizó en el desarrollo de un modelo de aprendizaje automático utilizado para el análisis de eficiencia energética en edificios residenciales, para lo cual se hizo uso de un conjunto de datos de rendimiento energético, que se encuentra disponible en el repositorio de la Universidad de California Irvine (UCI). El modelo de análisis es de tipo supervisado por lo que la base de datos se dividirá en un conjunto de datos para entrenamiento y uno para prueba del modelo; obteniendo así las predicciones de carga de calefacción y refrigeración, cabe recalcar que el método de aprendizaje de maquina será basado en técnicas de regresión lineal y siguiendo los seis pasos de creación de modelos de machine learning descritos por Manrique, (2020) y mencionado en el capítulo anterior.

2.1. Recolección de los datos.

Como primer paso para el desarrollo de esta investigación se iniciara con la recolección del conjunto de datos de rendimiento energético, el cual será extraído del repositorio de la Universidad de California Irvine (UCI) como se puede observar en la Figura 1-2, por lo cual una vez extraída los datos se procederá a buscar en diferentes fuentes bibliográficas información relevante que hable sobre técnicas de ML relacionada con la eficiencia energética en edificios residenciales lo cual será una base para la creación del modelo de aprendizaje automático y así definir el problema que se pretende resolver con este.



UCI Machine Learning Repository
Center for Machine Learning and Intelligent Systems

About Citation Policy Donate a Data Set Contact

Search

Repository Web

View ALL Data Sets

Check out the beta version of the new UCI Machine Learning Repository we are currently testing! Contact us if you have any issues, questions, or concerns. Click here to try out the new site.

Energy efficiency Data Set

Download [Data Folder](#) [Data Set Description](#)

Abstract: This study looked into assessing the heating load and cooling load requirements of buildings (that is, energy efficiency) as a function of building parameters.

Data Set Characteristics:	Multivariate	Number of Instances:	768	Area:	Computer
Attribute Characteristics:	Integer, Real	Number of Attributes:	8	Date Donated	2012-11-30
Associated Tasks:	Classification, Regression	Missing Values?	N/A	Number of Web Hits:	396341

Source:

The dataset was created by Angeliki Xifara (angxifara.1@gmail.com, Civil/Structural Engineer) and was processed by Athanasios Tsanas (tsanasthanasis.1@gmail.com, Oxford Centre for Industrial and Applied Mathematics, University of Oxford, UK).

Figura 1-2: Repositorio de la Universidad de California Irvine (UCI)

Fuente:(Irvine, 2012)

2.1.1. Definición del problema.

La base de datos de eficiencia energética con la cual se realizará el presente trabajo de integración curricular tiene como principal objetivo analizar y evaluar los requisitos de las cargas de calefacción y las cargas de refrigeración en edificios residenciales (es decir, la eficiencia energética) en función de los parámetros de los edificios por lo que se puede decir que existen dos enfoques diferentes para este análisis. Como primer enfoque se puede predecir las C.C y C.R. es decir el consumo energético que existe en un edificio; y como segundo enfoque se pueden realizar una clasificación donde se determine la calidad del consumo energético de acuerdo con la distribución de los parámetros de un edificio, debido a esto si en futuras investigaciones se quisiera abordar los dos enfoques en un mismo análisis se debería desarrollar dos modelos de aprendizaje de máquina , uno basado en técnicas de regresión y otro en clasificación.

Debido a todo lo mencionado anteriormente el análisis para esta investigación se orientará únicamente al primer enfoque, donde el principal problema será evaluar el consumo energético que existe en los diversos edificios residenciales y los requisitos para mejorar la eficiencia energética en base a sus cargas de calefacción y refrigeración, para esto se utilizará un conjunto de datos de rendimiento energético. Por lo que una vez analizado esta información se puede determinar los principales elementos del problema, los cuales son:

- Características del conjunto de datos: los datos se encuentran completo sin valores faltantes además de contar con 768 instancias multivariadas con 8 atributos de características de valor entero y valor real.
- Tarea asociada: regresión.
- Destino: predicción de las cargas de calefacción y las cargas de refrigeración de los diferentes edificios residenciales (es decir, la eficiencia energética) en función de los parámetros de cada edificio.

2.2. Preprocesamiento del conjunto de datos.

Como segundo paso se llevará a cabo el preprocesamiento del conjunto de datos en donde se realizará: la descripción del conjunto de datos, importación de librerías necesarias para la creación del modelo de ML, la lectura de los datos extraídos del repositorio, una preparación previa a los datos con el cambio de nombre de las columnas para un mejor entendimiento así como la comprobación y la limpieza de los datos , para lo cual se utilizara el software de gestión de aplicaciones Anaconda Navigator como se ve en la Figura 2-2, en especial se utilizó la aplicación Jupyter Notebook la cual es una aplicación basada en la web, utilizada principalmente para la

carga de datos, visualización, estadísticas, procesamiento de imágenes entre otras, además que es una aplicación compatible con el lenguaje de programación Python.

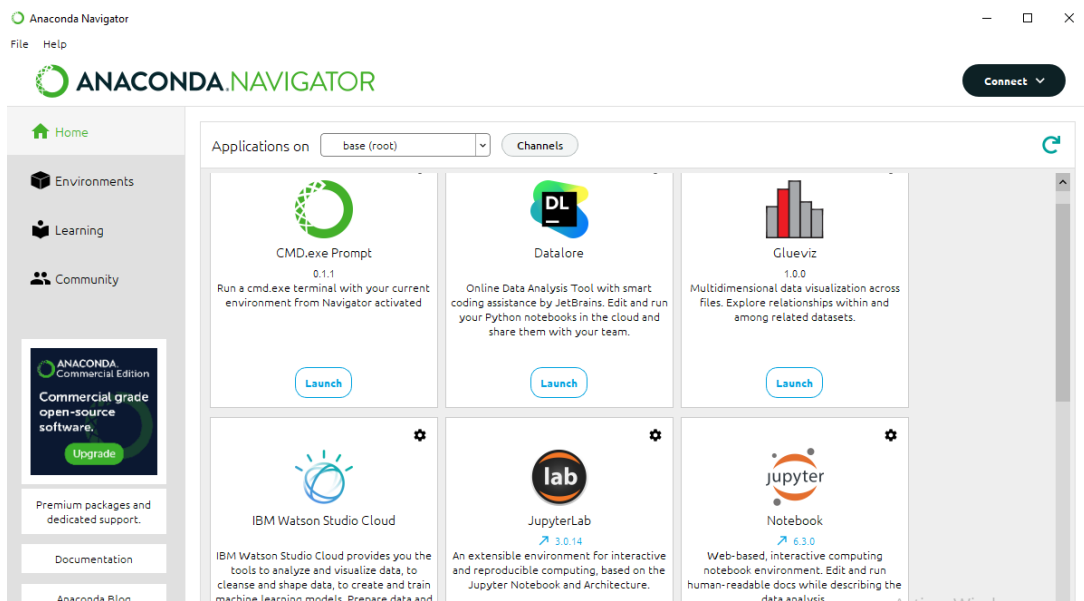


Figura 2-2: Software Anaconda navigator

Fuente: (Anaconda software, 2020)

2.2.1. Descripción general del conjunto de datos.

El conjunto de datos de rendimiento energético que se obtuvo del repositorio está conformado por una carpeta llamada (EnergyEfficiency) la cual contiene un archivo EXCEL en formato CSV, correspondiente a un análisis de energía de 12 formas de edificios diferentes. Los datos de los edificios se diferencian en ocho atributos o características, denotados por X1 a X8 y dos respuestas o resultados, denotados por Y1 y Y2. Por lo que se puede analizar para realizar predicciones de carga de calefacción y refrigeración, es decir evaluar la eficiencia energética en función de los parámetros de los edificios.

De manera más específica los atributos del conjunto están conformados de la siguiente forma:

X1 Compacidad relativa.

X2 Superficie - m².

X3 Superficie de la pared - m².

X4 Superficie del techo - m².

X5 Altura total – m.

X6 Orientación - 2: Norte, 3: Este, 4: Sur, 5: Oeste.

X7 Área de acristalamiento: 0%, 10%, 25%, 40% (del área del piso).

X8 Distribución del área de acristalamiento (variación) - 1: uniforme, 2: norte, 3: este, 4: sur, 5: oeste.

Y1 Carga de calefacción – kWh.

Y2 Carga de refrigeración – kWh.

Cabe mencionar que la base de datos mencionada es una simulación hecha en el software Ecotec que cuenta con varios escenarios en función de las características antes mencionadas con lo que se obtiene 768 formas de edificios con el objetivo de predecir dos respuestas de valor real.

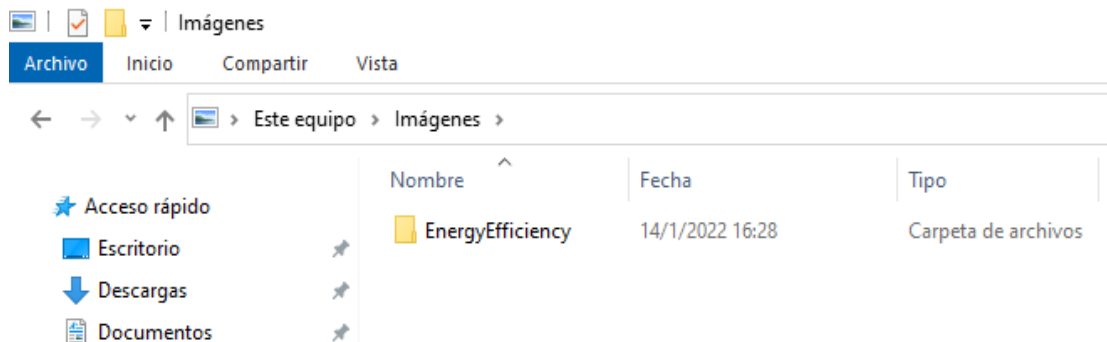


Figura 3-2: Conjunto de datos de eficiencia energética.

Realizado por: Charanchi J. 2022

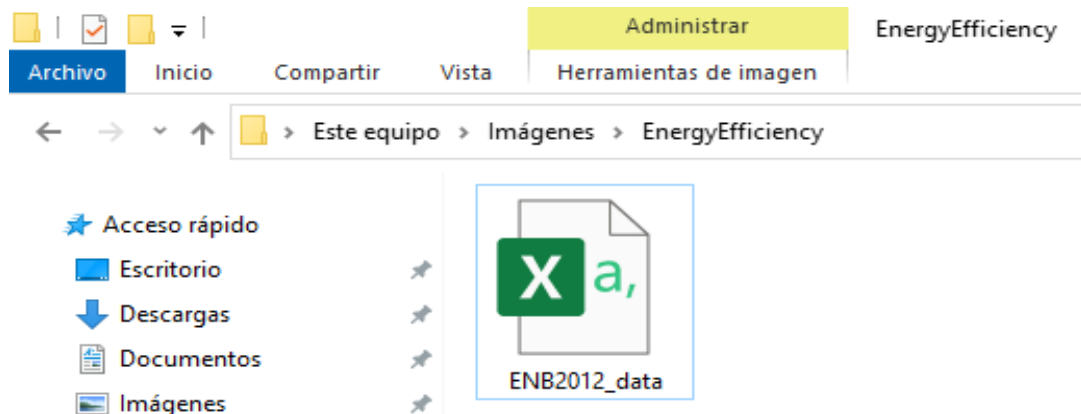


Figura 4-2: Conjunto de datos de eficiencia energética.

Realizado por: Charanchi J. 2022

2.2.2. *Importación de librerías.*

Una vez se haya descrito el conjunto de datos y determinado las características que se utilizarán para la creación del modelo de ML se procederá con la importación de todas las librerías necesarias que se utilizarán para la lectura, la visualización, manejo y cálculo del conjunto de datos, estas librerías son pandas, numpy, matplotlib, scipy, entre otras.

Cabe mencionar que estas librerías no se cargan de manera predeterminada, por lo que se deben ser importadas, y en algunas ocasiones deberán ser instaladas debido a que no vienen incluidas en el paquete de Python. como se muestra en el siguiente código.

```
#Importación de todas las bibliotecas
import numpy as np
import pandas as pd
from sklearn.preprocessing import StandardScaler

#construcción del modelo
import statsmodels.api as sm
import statsmodels.formula.api as smf

#evaluación del modelo
from sklearn.feature_selection import RFE
from statsmodels.stats.outliers_influence import
variance_inflation_factor
from sklearn import metrics
from sklearn.metrics import precision_recall_curve
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import explained_variance_score
from sklearn.metrics import mean_squared_error
from sklearn.metrics import r2_score

#validación del modelo
from sklearn.model_selection import train_test_split

#visualización
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

# Supresión de advertencias
import warnings
warnings.filterwarnings('ignore')

# Personalización de la visualización de datos
pd.set_option('display.max_columns', 100)
```

2.2.3. Lectura del conjunto de datos.

Para la lectura del conjunto de datos de eficiencia energética se la realizó utilizando la librería Pandas que es una librería especializada en el análisis y manejo de estructuras de datos, la función que se utilizó es `read_csv()` que permite cargar los archivos en un tipo de objeto llamado DataFrame. En total se realizó la lectura de 1 archivo de Excel con extensión csv. el cual contenía 10 columnas con 768 filas. y fue nombrado con el nombre como se muestra en el siguiente código

```
# lectura del conjunto de datos de EFICIENCIA ENERGETICA
datos=pd.read_csv("C:/Users/USUARIO/Pictures/EnergyEfficiency/ENB2012_
data.csv")
```

En este apartado también se renombraron las columnas, debido a que los nombre con los que contaba el conjunto de datos denotaba a los ocho atributos o características como: X1 hasta X8 y las dos respuestas o resultados, como Y1 y Y2 por lo que no se brindaba la información necesaria para un buen entendimiento de estos por lo cual se utilizó la esta función `datos.columns` que reemplaza los nombres originales por nombres que brinden la información necesaria como se muestra en el siguiente código.

```
# Renombramos las columnas
datos.columns=['Compacidad_relativa', 'Superficie', 'Superficie_de_la_pared', 'Superficie_del_techo', 'Altura_total', 'Orientación', 'Área_de_acristalamiento', 'Distribución_del_área_de_acristalamiento', 'Carga_de calefacción', 'Carga_de_refrigeración']
```

2.2.4. Conjunto de datos de eficiencia energética.

Tabla 1-2: Conjunto de datos de eficiencia energética.

Capacidad relativa	Superficie	Superficie de la pared	Superficie del techo	Altura total	Orientación	Área de acristalamiento	Distribución del área de acristalamiento	Carga de calefacción	Carga de refrigeración
0.98	514.5	294.0	110.25	7.0	2	0.0	0	15.55	21.33
0.98	514.5	294.0	110.25	7.0	3	0.0	0	15.55	21.33
0.98	514.5	294.0	110.25	7.0	4	0.0	0	15.55	21.33
0.98	514.5	294.0	110.25	7.0	5	0.0	0	15.55	21.33
0.90	563.5	318.5	122.50	7.0	2	0.0	0	20.84	28.28
...
0.64	784.0	343.0	220.50	3.5	5	0.4	5	17.88	21.40
0.62	808.5	367.5	220.50	3.5	2	0.4	5	16.54	16.88
0.62	808.5	367.5	220.50	3.5	3	0.4	5	16.44	17.11
0.62	808.5	367.5	220.50	3.5	4	0.4	5	16.48	16.61
0.62	808.5	367.5	220.50	3.5	5	0.4	5	16.64	16.03

Realizado por: Charanchi, Javier, 2022

2.2.5. Limpieza de los datos.

La limpieza de datos es uno de los pasos más importantes y fundamentales a la hora de preparar los datos antes de crear un modelo de ML debido a que se pueden detectar múltiples errores que pueden reducir la eficacia del modelo por lo que detectarlos en este paso y corregirlos, aumenta la efectividad y el rendimiento de un modelo predictivo bajo un enfoque de ML, en otras palabras se puede decir que dependiendo de la calidad de los datos y de que si estos son correctamente seleccionados, limpiados y transformados mejoran un algoritmo dándonos predicciones de muy buena calidad.

Para el manejo de valores faltantes en el conjunto de datos se utilizó la función `isna().sum()`, que realiza una búsqueda de la existencia de valores faltantes o nulos, devolviéndonos el número de valores faltantes en cada columna como se muestra en el siguiente código.

```
# comprobación de valores faltantes
datos.isna().sum()

Capacidad relativa          0
Superficie                  0
Superficie de la pared     0
Superficie del techo       0
Altura total                0
Orientación                 0
Área de acristalamiento    0
Distribución del área de acristalamiento 0
Carga de calefacción       0
Carga de refrigeración     0
dtype: int64
```

Como se puede observar en el código para la comprobación de valores faltantes, el resultado que muestra el programa refleja que no existen valores faltantes en ninguna de las columnas, por lo que se puede determinar que el conjunto de datos se encuentra completo, cabe recalcar que no todas las bases de datos presentan este caso por lo que, dependiendo el análisis del estudio, los datos faltantes pueden ser eliminarlos o remplazarlos, siempre que se tenga en cuenta que los datos son el activo más valioso a la hora de crear un modelo de ML, debido a que los modelos de aprendizaje automático funcionan mejor cuando más datos existen y no se deben descartar tan fácilmente, qué es preferible remplazarlos con estimaciones estadísticas como la media, mediana y moda que es el método que comúnmente se utiliza.

2.3. Análisis exploratorio de los datos.

Como tercer paso se llevará a cabo un análisis exploratorio del conjunto de datos en donde se realizará: una búsqueda de la información del conjunto de datos, una descripción estadística, un análisis gráfico y un análisis de correlación para obtener toda la información y poder visualizar la disposición de los datos con respecto a las variables resultantes que se espera obtener con el modelo de ML.

2.3.1. Búsqueda de la información del conjunto de datos.

En este apartado se realizó una búsqueda de toda la información del conjunto de datos para determinar que no existan datos categóricos debido a que los modelos de ML se basan únicamente en ecuaciones y cálculos matemáticos por lo que todos los datos que se utilizaran deberán ser de

tipo numérico. Para este propósito se utilizó la función *info()* de pandas, que imprime una lista resumida y concisa de los datos, donde se puede visualizar el tipo de datos, el tipo de columna, el número de filas, los valores no nulos y el uso de la memoria que hacen estos. Como se muestra a en el código a continuación, no se encontraron datos categóricos, todos los datos corresponden únicamente a valores enteros int64 y de tipo numéricos float 64.

Así también es importante recalcar que, si en el conjunto de datos hubiese existido valores categóricos, estos valores deberás ser categorizados a números enteros para asegurar que el algoritmo interprete correctamente las características originales de estos datos y en caso de que información de estos no sea indispensable para la creación del modelo de ML se recomienda eliminarlos.

```
# Imprimimos la información de los datos que contiene del conjunto de datos
```

```
datos.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 10 columns):
#      Column                                     Non-Null Count  Dtype
---  -
0 Capacidad relativa                          768 non-null    float64
1 Superficie                                  768 non-null    float64
2 Superficie de la pared                      768 non-null    float64
3 Superficie del techo                        768 non-null    float64
4 Altura total                                768 non-null    float64
5 Orientación                                 768 non-null    int64
6 Área de acristalamiento                    768 non-null    float64
7 Distribución del área de acristalamiento   768 non-null    int64
8 Carga de calefacción                       768 non-null    float64
9 Carga de refrigeración                     768 non-null    float64

dtypes: float64(8), int64(2)
memory usage: 60.1 KB
```

2.3.2. Descripción estadística de los datos.

La descripción estadística de los datos es uno de los pasos primordiales a la hora de explorar el conjunto de datos debido a que podemos ver en detalles parámetros estadísticos como percentil, la media, la desviación estándar, el valor mínimo y máximo de nuestro conjunto de datos, así también como cuántos datos se tiene. Estos datos por lo general se utilizan para analizar información esencial que será usada para el análisis de datos y la derivación de diferentes supuestos matemáticos para futuras investigaciones.

Para este apartado se utilizó la función *describe()* de pandas, que calcula los datos antes mencionados e imprime una lista de estos como se puede mostrar en el siguiente código y en la Tabla 2-2.

```
# Descripción estadística de los datos
datos.describe().transpose()
```

Tabla 2-2: Descripción estadística del conjunto de datos de eficiencia energética.

Variables	count	mean	std	min	25%	50%	75%	max
Compacidad relativa	768.0	0.764167	0.105777	0.62	0.6825	0.75	0.8300	0.98
Superficie	768.0	671.708333	88.086116	514.50	606.3750	673.75	741.1250	808.50
Superficie de la pared	768.0	318.500000	43.626481	245.00	294.0000	318.50	343.0000	416.50
Superficie del techo	768.0	176.604167	45.165950	110.25	140.8750	183.75	220.5000	220.50
Altura total	768.0	5.250000	1.751140	3.50	3.5000	5.25	7.0000	7.00
Orientación	768.0	3.500000	1.118763	2.00	2.7500	3.50	4.2500	5.00
Área de acristalamiento	768.0	0.234375	0.133221	0.00	0.1000	0.25	0.4000	0.40
Distribución del área de acristalamiento	768.0	2.812500	1.550960	0.00	1.7500	3.00	4.0000	5.00
Carga de calefacción	768.0	22.307201	10.090196	6.01	12.9925	18.95	31.6675	43.10
Carga de refrigeración	768.0	24.587760	9.513306	10.90	15.6200	22.08	33.1325	48.03

Realizado por: Charanchi, Javier, 2022

2.3.3. Análisis gráfico.

En la exploración del conjunto de datos una de las herramientas más utilizadas y de mayor importancia son los gráficos de barras o histogramas, debido a que estas nos ayudan a visualizar la cantidad de datos existentes para cada una de las variables, además de ayudarnos a determinar si nuestros datos siguen una distribución normal o dependiendo de la distorsión de la curva de la distribución normal en los datos presentan un sesgo positivo o negativo

Para este apartado se utilizó la función *hist()* de la librería NumPy, la cual calcula y dibuja un histograma que nos permite visualizar la distribución de los datos.

Como se puede observar en la Figura 5-2 las distribuciones que presentan las variables de entrada y salida demuestran que ninguna de estas variables sigue una distribución normal por lo que todas las variables deben ser analizadas para la predicción.

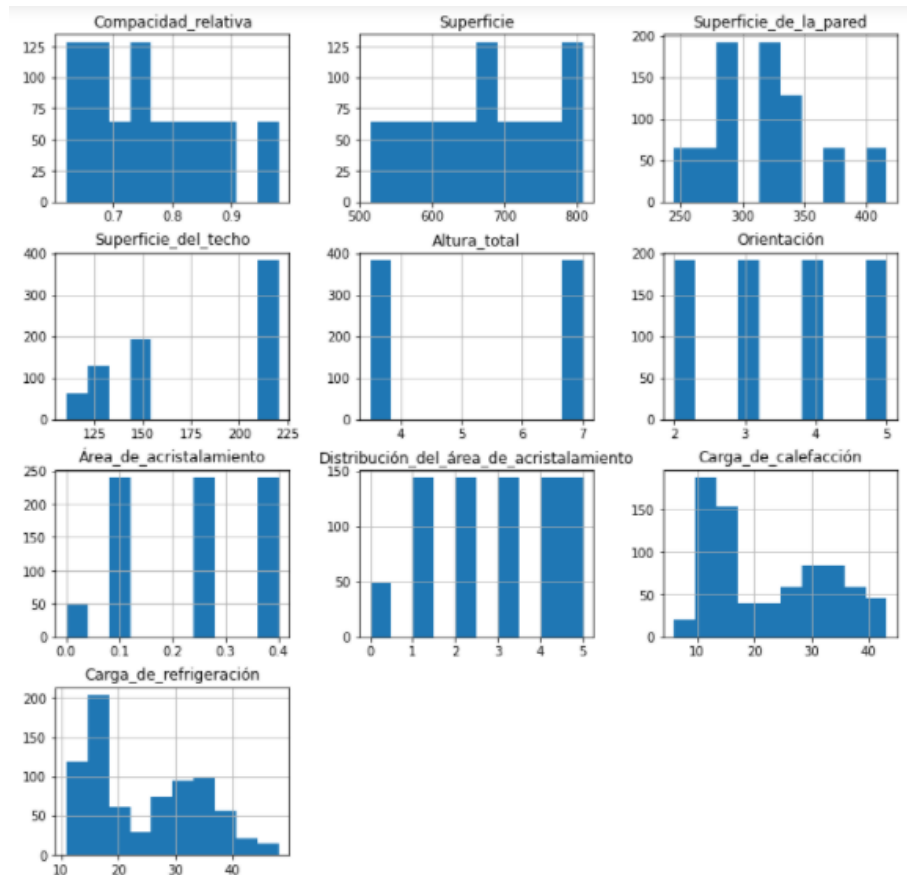


Figura 5-2: Descripción estadísticas del conjunto de datos

Realizado por: Charanchi J. 2022

2.3.4. *Análisis de correlación.*

Como se pudo observar en los histogramas y los diversos diagramas de barras y cajas las dos variables de respuesta están altamente correlacionadas entre sí. Esto significa que esperamos ver un comportamiento similar en su distribución entre sus diferentes clases de tamaño.

Para este apartado se utilizó la función. `corr()` de Python, esta función calcula el coeficiente de correlación producto-momento de Pearson entre nuestros datos. Esto prácticamente significa que al aplicar esta función podemos obtener una visión sólida de cómo todos y cada uno de los atributos de nuestro conjunto de datos se correlacionan entre sí.

Un coeficiente de correlación es un número que cuantifica algún tipo de correlación y dependencia, es decir, relaciones estadísticas entre dos o más variables aleatorias o valores de datos observados por lo que se puede decir que en la mayoría de los modelos de ML, se ven afectados cuando las variables están altamente correlacionadas, por lo que es conveniente estudiar el grado de correlación entre estas. En el gráfico 6-2, se muestra el grado de correlación entre las variables del conjunto de datos y en este se puede apreciar que la correlación positiva más alta es

de (0,90) y la correlación negativa más baja es de (-0,90), estos valores son cercanos a cero, lo que indica una correlación muy débil entre las variables, por lo tanto, no están correlacionadas linealmente.

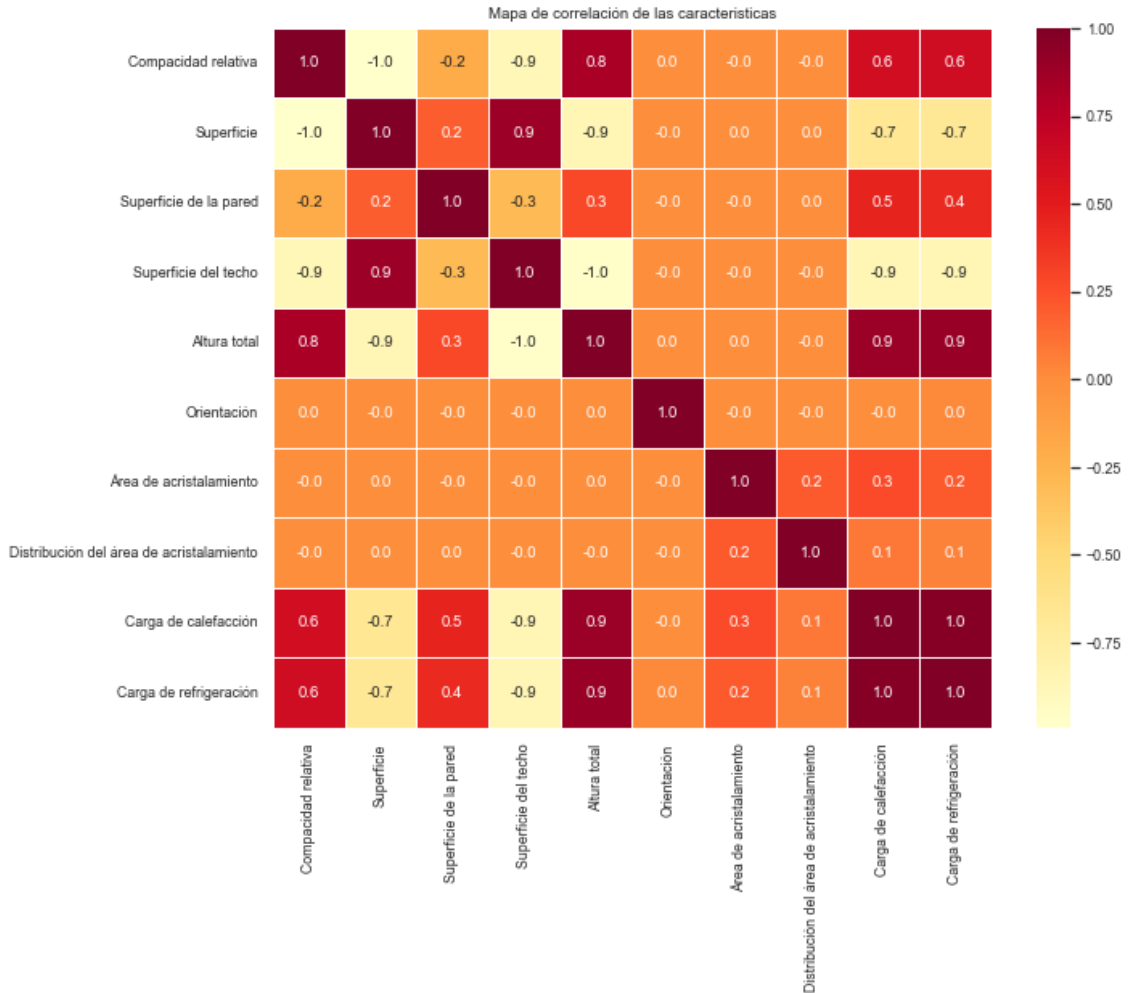


Figura 6-2: Mapa de correlaciones del conjunto de datos

Realizado por: Charanchi J. 2022

Cabe mencionar que en la Figura 6-2, la matriz de correlación nos indica que existe una fuerte correlación entre los objetivos y que no se puede excluir uno de ellos, debido a que las cargas de calefacción y las cargas de refrigeración son salidas igualmente importantes que se deben predecir.

También podemos observar la información sobre las correlaciones entre todas las variables. Por ejemplo, la altura general (una entrada) tiene una fuerte correlación (0,90) con la salida: carga de refrigeración. Además, el diagrama de pares representa la relación entre ellos. Para la gráfica de altura general y carga de refrigeración, solo hay 2 valores de altura general debido a la distribución

y nos dificulta ver las correlaciones lineales de esas variables. Usaremos el método de preprocesamiento para refinar las distribuciones.

2.4. Entrenamiento del modelo.

2.4.1. *Normalizado del conjunto de datos.*

La normalización de un conjunto de datos es de suma importancia en los modelos de ML debido a que la mayoría de los algoritmos, requieren que todas las características estén normalizadas o escaladas con la finalidad de que ninguna de las características presentes en la base de datos se confunda con otra. Para este apartado se utilizó la función *Normalizer* de Python, esta función ayuda a minimizar el impacto o errores que se pueden dar en los resultados por valores atípicos como se muestra en el siguiente código. es decir, se crea un objeto normalizador donde se ajusta sus parámetros entre los cuales están la media y la desviación estándar.

```
#Normaliza las entradas y divide la salida
from sklearn.preprocessing import Normalizer
nr = Normalizer(copy=False)
```

2.4.2. *División del conjunto de datos.*

Para la división del conjunto de datos en los diferentes algoritmos se decidió dividirlo dos partes: una parte del conjunto de datos para la fase de entrenamiento y la otra parte para la fase de prueba del modelo, por lo que una de las formas más común y más utilizada por los investigadores a la hora de dividir los datos es 75% para la fase entrenamiento y 25% para la fase de prueba, debido a esto los datos de eficiencia energética empleados en la presente investigación, serán divididos de esa manera tal y como se muestra en el siguiente código.

```
# División del conjunto de datos para prueba y emtenamiento
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size =
0.25, random_state = 10)

print('X Train:',x_train.shape)
print('X Test:',x_test.shape)
print('Y Train:',y_train.shape)
print('Y Test:',y_test.shape)

X Train: (576, 8)
X Test: (192, 8)
Y Train: (576, 2)
Y Test: (192, 2)
```

2.4.3. Creación del modelo de ML.

Una vez normalizado y dividido los datos para entrenamiento y prueba, se procedió con la creación de los diversos modelos de ML para el análisis de eficiencia energética en edificios residenciales para lo cual se utilizó la biblioteca Sklearn de Python la cual nos ayuda a llamar a los diferentes modelos y las herramientas necesarias para las diversas predicciones, tal y como se muestra en el siguiente código.

```
# carga del modelo de regresión
from sklearn.linear_model import LinearRegression

# crea el modelo de regresión lineal
Regresión = LinearRegression()

# Ajuste de los datos de (x) y (y) para la regresión
Regresión.fit(x_train, y_train)

# Predice la respuesta del modelo
y_pred1 = Regresión.predict(x_test)
```

Una vez cargada la biblioteca con modelo de regresión que se decidió a utilizar, se procede a crear un modelo de regresión sin optimizar para determinar si este modelo cumple con las condiciones necesarias para una predicción exacta, así también se procede con un ajuste de los datos y la evaluación para cada uno de los modelos, siendo la más importante, la métrica del r2 cuadrado.

2.5. Validación del modelo.

Una de las validaciones que se utilizó para los diversos modelos fue el R2 cuadrado debido a que esta nos ayuda a determinar la precisión de las predicciones hechas por los diversos modelos.

Así teniendo en cuenta la distribución y la correlación en nuestra exploración de los datos, usamos bien todos los valores, ya que queremos crear el mejor ajuste para las líneas de predicción mediante la evaluación de R cuadrado. Siempre aumentará a medida que agreguemos más variables independientes. Para obtener una R alto, incluimos todas las variables.

2.5.1. Predicción del conjunto de datos.

Para comprobar las diversas predicciones realizadas por los modelos se decidió realizar una comparativa imprimiendo los valores de las cargas de calefacción y refrigeración que nos brindaba el conjunto de datos y los valores que nos arrojaba el modelo como se muestra en los siguientes códigos.

```

#Predicción para la carga de calefacción
Predicción_calefacción1 = pd.DataFrame({'Valor Actual':
y_test['Carga_de_calefacción'], 'Predicción': y_pred1[:,0]})
predicción_calefacción1

#Predicción para la carga de calefacción
predicción_refrigeración1 = pd.DataFrame({'Valor Actual':
y_test["Carga_de_refrigeración"], 'Predicción': y_pred1[:,1]})
predicción_refrigeración1

```

2.5.2. Métricas para la evaluación del modelo.

Entre las múltiples métricas que se utilizó para la evaluación de los modelos de predicción están el error medio cuadrado y el error absoluto medio, que nos presentan valores muy elevados demostrando que las presiones del modelo y la calidad de las predicciones son de excelente calidad, tal y como se muestra en el siguiente código.

```

# Métricas

print('Error absoluto medio (MAE):', mean_absolute_error(y_test,
y_pred1, sample_weight=None, multioutput='uniform_average'))

print('Desviación:', explained_variance_score(y_test, y_pred1,
multioutput='variance_weighted'))

print('Error cuadrático medio (MSE):', mean_squared_error(y_test,
y_pred1, sample_weight=None, multioutput='uniform_average'))

print('Raíz del Error cuadrático medio
(RMSE):', np.sqrt(mean_squared_error(y_test, y_pred1,
sample_weight=None, multioutput='uniform_average')))

print('R cuadrado:', r2_score(y_test, y_pred1, sample_weight=None,
multioutput='uniform_average'))

```

CAPÍTULO III

3. RESULTADOS Y DISCUSIÓN.

3.1. Interpretación de los resultados obtenidos en cada método experimental.

El presente capítulo tiene como objetivo principal analizar e interpretar los resultados obtenidos luego de la culminación del desarrollo de los diversos modelos de aprendizaje automático para la predicción de la eficiencia energética en edificios residenciales.

3.2. Modelo experimental 1 basado en Regresión Lineal.

Como paso número uno para la construcción de modelo experimental 1 basado en Regresión lineal se importó las librerías necesarias utilizando la biblioteca de sklearn una vez importada las librerías se procedió a la creación del modelo en cuyo caso se lo nombro regresión, así creado el modelo se lo entreno para que realice las predicciones antes mencionadas.

Así también se imprimió la precisión que nos arroja el modelo en este caso siendo del 92,07%, tal y como se muestra en el siguiente código.

Regresión lineal

```
# carga el modelo de regresión
from sklearn.linear_model import LinearRegression
# crea el modelo de regresión lineal
Regresión = LinearRegression()
# Ajuste de los datos de (x) y (y) para la regresión
Regresión.fit(x_train, y_train)
# Predice la respuesta del modelo
y_pred1 = Regresión.predict(x_test)
acc_linreg = round(Regresión.score(x_test,y_test)* 100, 2)
print("Precisión del modelo de regresión lineal:",acc_linreg,"%")
Precisión del modelo de regresión lineal: 92.07 %
```

3.2.1. Métricas de evaluación del modelo experimental 1.

```
Error absoluto medio (MAE): 1.879675202546295
Desviación: 0.9217301345045793
Error cuadrático medio (MSE): 7.243364168497955
Raíz del Error cuadrático medio (RMSE): 2.691349878499255
R cuadrado: 0.9206741725373426
```

A continuación, se dará a conocer el resultado de las métricas de evaluación:

- El error absoluto medio (MAE): es de 1,87, es decir este valor representa la distancia a la cual se encuentra el valor de que la predicción con respecto al valor real, por lo que se puede decir que el resultado de la predicción no esta tan alejada al valor que nos da en el conjunto de datos.
- La Desviación: es de 0,92, este valor representa el error que puede tener el valor predicho con el valor que se nos dio en el conjunto de datos al estar tan cercano a 1 se puede considerar que el valor predicho si es fiable.
- El error cuadrático medio (MSE): es de 7,24, este valor representa la medida de qué tan cerca está una línea ajustada de los puntos de los datos por lo que se puede decir que varios valores predichos se encuentran muy lejanos con respecto a los valores reales debido a esto se puede decir que el modelo presentara errores en sus predicciones.
- La raíz del error cuadrático medio (RMSE): es de 2,69, este valor representa la distancia, en promedio, de un punto de datos de la línea ajustada, medida a lo largo de una línea vertical por lo que se puede decir que el residuo es muy grande.
- El R2 cuadrado: es de 0,92, este valor representa el porcentaje de variación que existe entre la variable de respuesta y su relación con una o más variables predictoras por lo que al estar cercana al 100% nos determina que la predicción del modelo es muy eficaz.

Así también para conocer de manera grafica la precisión de evaluación del mejor modelo obtenido en el ME1 y comprobar que tan fiables son los resultados obtenidos, se aplicó la técnica de validación cruzada (CV), para obtener una aproximación más real de la precisión del modelo y a su vez para conocer con que precisión puede generalizar nuevos datos por lo que con la ayuda de un gráfico de dispersión se puede visualizar que tan cercanos son los datos predichos a una recta normal tal y como se muestra en el siguiente código.

```
from sklearn.model_selection import cross_val_predict

# Genera estimaciones de la validación cruzada para cada punto de
datos de entrada

predicted = cross_val_predict(Regresión, x, y, cv=10)

# Traza las salidas previstas con respecto a las medidas

fig, ax = plt.subplots()
ax.scatter(y, predicted)
ax.plot([y.min(), y.max()], [y.min(), y.max()], 'k--', lw=4)
ax.set_xlabel('Valor medido')
ax.set_ylabel('Predicción')
plt.show()
```

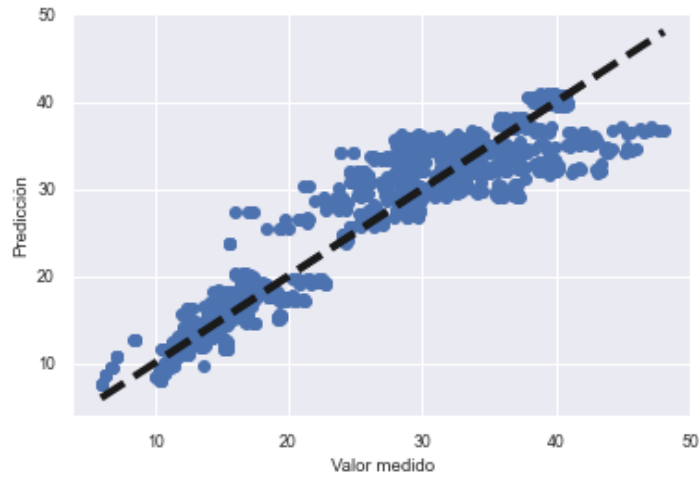


Figura 1-3: Precisión del modelo experimental 1 basado en regresión lineal

Realizado por: Charanchi J. 2022

3.2.2. Comparación de los valores del conjunto de datos y los predichos por el modelo.

Una vez realizada la predicción de las CC y CR se pudo determinar que los valores predichos por el modelo son muy cercanos a los valores reales del conjunto de datos por lo que se pueden tomar como fiables para los diferentes estudios de análisis.

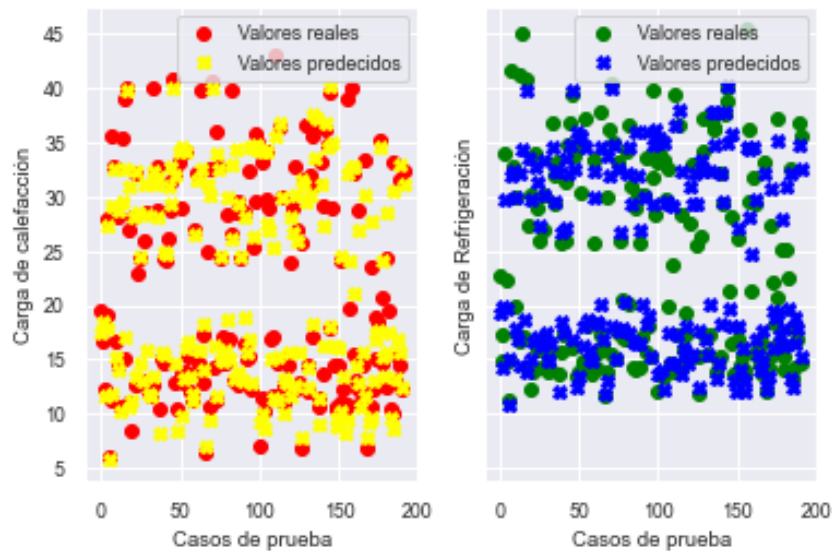


Figura 2-3: Comparación de los valores predichos del modelo experimental 1

Realizado por: Charanchi J. 2022

3.3. Modelo experimental 2 basado en Decisión Tree Regressor.

Como paso número uno para la construcción de modelo experimental 2 basado en Decision Tree Regressor se importó las librerías necesarias utilizando la biblioteca de sklearn una vez importada

las librerías se procedió a la creación del modelo en cuyo caso se lo nombro dt_model , así creado el modelo se lo entreno para que realice las predicciones antes mencionadas.

Así también se imprimió la precisión que nos arroja el modelo en este caso siendo del 96,65%, tal y como se muestra en el siguiente código.

Decision Tree Regressor

```
#Carga el modelo de decision tree regressor
from sklearn.tree import DecisionTreeRegressor
# crea el modelo de decision tree regressor
dt_model = DecisionTreeRegressor(random_state=123)
# Ajuste de los datos de (x) y (y) para la regresión
dt_model.fit(x_train, y_train)
# Predice la respuesta del modelo
y_pred2 = dt_model.predict(x_test)
acc_dtr = round(dt_model.score(x_test,y_test)* 100, 2)
print(" Precisión del modelo de regresion de decision tree regressor
:" ,acc_dtr,"%")
Precisión del modelo de regresion de decision tree regressor : 96.65 %
```

3.3.1. Métricas de evaluación del modelo experimental 2.

```
Error absoluto medio (MAE): 0.8427604166666666
Desviación: 0.9683323767600238
Error cuadrático medio (MSE): 2.9239031250000003
Raíz del Error cuadrático medio (RMSE): 1.7099424332415405
R cuadrado: 0.966513701581212
```

A continuación, se dará a conocer el resultado de las métricas de evaluación:

- El error absoluto medio (MAE): es de 0,84, es decir este valor representa la distancia a la cual se encuentra el valor de que la predicción con respecto al valor real, por lo que se puede decir que el resultado de la predicción no esta tan alejada al valor que nos da en el conjunto de datos.
- La Desviación: es de 0,96, este valor representa el error que puede tener el valor predicho con el valor que se nos dio en el conjunto de datos al estar tan cercano a 1 se puede considerar que el valor predicho si es fiable.
- El error cuadrático medio (MSE): es de 2,92, este valor representa la medida de qué tan cerca está una línea ajustada de los puntos de los datos por lo que se puede decir que varios de los valores predichos son lejanos a los reales por lo que el modelo presentara errores leves en sus predicciones.

- La raíz del error cuadrático medio (RMSE): es de 1,70, este valor representa la distancia, en promedio, de un punto de datos de la línea ajustada, medida a lo largo de una línea vertical por lo que se puede decir que el residuo es muy grande.
- El R2 cuadrado: es de 0,96, este valor representa el porcentaje de variación que existe entre la variable de respuesta y su relación con una o más variables predictoras por lo que al estar cercana al 100% nos determina que la predicción del modelo es muy eficaz.

Así también para conocer de manera grafica la precisión de evaluación del mejor modelo obtenido en el ME2 y comprobar que tan fiables son los resultados obtenidos, se aplicó la técnica de validación cruzada (CV), para obtener una aproximación más real de la precisión del modelo y a su vez para conocer con que precisión puede generalizar nuevos datos por lo que con la ayuda de un gráfico de dispersión se puede visualizar que tan cercanos son los datos predichos a una recta normal tal y como se muestra en el siguiente código.

```
from sklearn.model_selection import cross_val_predict

# Genera estimaciones de la validación cruzada para cada punto de
datos de entrada
predicted = cross_val_predict(dt_model, x, y, cv=10)

# Traza las salidas previstas con respecto a las medidas
fig, ax = plt.subplots()
ax.scatter(y, predicted)
ax.plot([y.min(), y.max()], [y.min(), y.max()], 'k--', lw=4)
ax.set_xlabel('Valor medido')
ax.set_ylabel('Predicción')
plt.show()
```

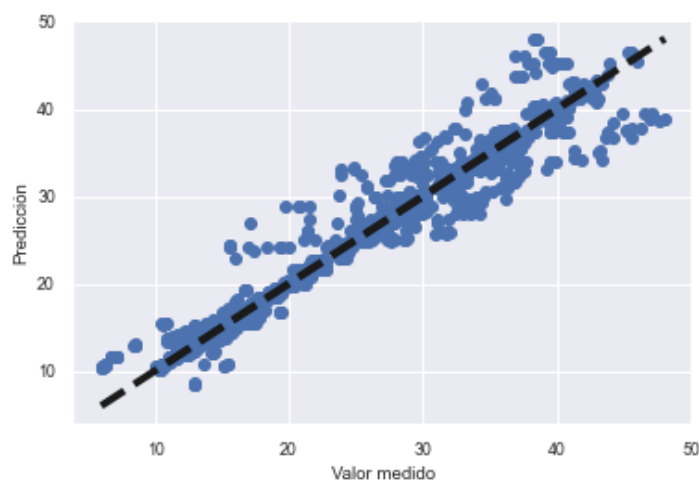


Figura 3-3: Precisión del modelo experimental 2

Realizado por: Charanchi J. 2022

3.3.2. Comparación de los valores del conjunto de datos y los predichos por el modelo.

Una vez realizada la predicción de las CC y CR se pudo determinar que los valores predichos por el modelo son muy cercanos a los valores reales del conjunto de datos por lo que se pueden tomar como fiables para los diferentes estudios de análisis.

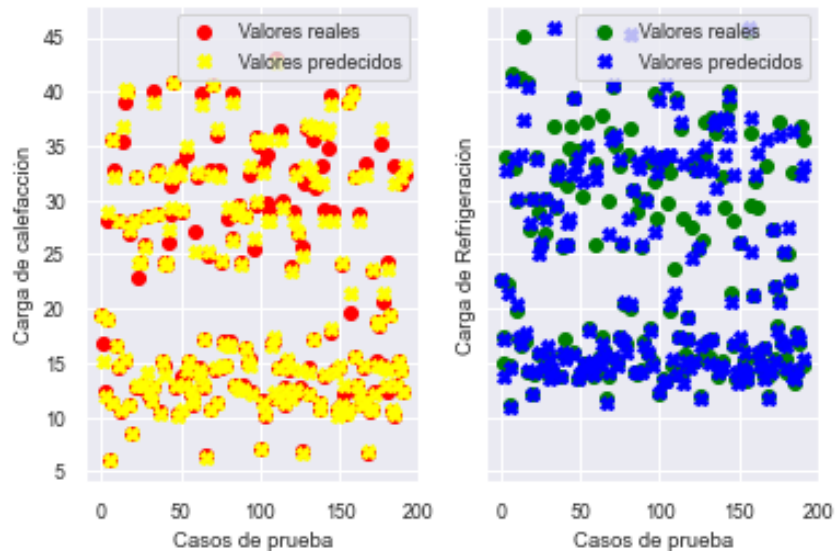


Figura 4-3: Comparación de los valores predichos del modelo experimental 2

Realizado por: Charanchi J. 2022

3.3.3. Optimización del modelo experimental 2.

Para la optimización del modelo experimental 2 se decidió encontrar los mejores parámetros para el rendimiento de este modelo, en este caso los parámetros a mejorar son la máxima profundidad que tiene el árbol, la mínima división que pueden tener las muestras y el mínimo de muestras de hojas que se pueden generar, tal y como se muestra en el siguiente código.

```
# Encuentra los mejores parámetros de optimización para el modelo de  
decision tree regressor
```

```
f, (ax1, ax2) = plt.subplots(1, 2, sharey=True)  
  
# Max Profundidad  
dt_acc = []  
dt_depth = range(1,11)  
for i in dt_depth:  
    dt = DecisionTreeRegressor(random_state=123, max_depth=i)  
    dt.fit(x_train, y_train)  
    dt_acc.append(dt.score(x_test, y_test))  
ax1.plot(dt_depth, dt_acc)  
ax1.set_title('Max Profundidad')
```

```

# Min División de muestras
dt_acc = []
dt_samples_split = range(10,21)
for i in dt_samples_split:
    dt = DecisionTreeRegressor(random_state=123, min_samples_split=i)
    dt.fit(x_train, y_train)
    dt_acc.append(dt.score(x_test, y_test))
ax2.plot(dt_samples_split,dt_acc)
ax2.set_title('Min División de muestras')

plt.show()

# Min Muestras de hoja
plt.figure(figsize = (5,5))
dt_acc = []
dt_samples_leaf = range(1,10)
for i in dt_samples_leaf:
    dt = DecisionTreeRegressor(random_state=123, min_samples_leaf=i)
    dt.fit(x_train, y_train)
    dt_acc.append(dt.score(x_test, y_test))

plt.plot(dt_samples_leaf,dt_acc)
plt.title('Min Muestras de hoja')

plt.show()

```

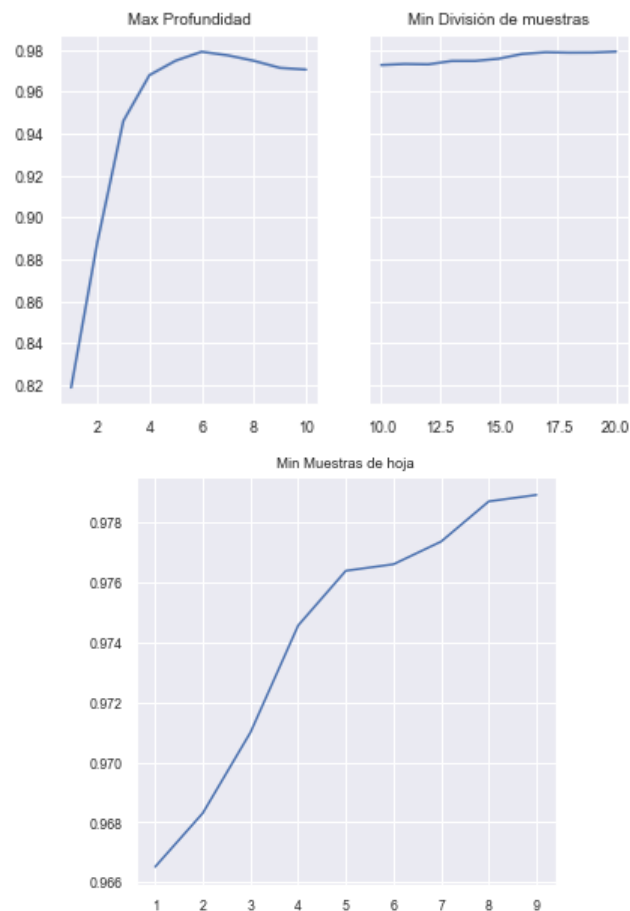


Figura 5-3: Mejores parámetros del modelo experimental 2

Realizado por: Charanchi J. 2022

Como se puede observar en la Figura 10-3 se procedió buscando los mejores parámetros en el conjunto de datos para luego decidir mediante graficas cual es el mejor valor que tomar en cuenta y predecir las variables respuestas.

3.3.4. Métricas de evaluación del modelo experimental 2 después de la optimización.

```
Error absoluto medio (MAE): 0.8215576987350885
Desviación: 0.9797979999820037
Error cuadrático medio (MSE): 1.8618527332313217
Raíz del Error cuadrático medio (RMSE): 1.3644972455931605
R cuadrado: 0.9789169597062091
```

Como se muestra en el código anterior, las métricas que arroja el modelo después de optimizar los parámetros antes definidos son mejores con respecto al modelo sin optimizar por lo que se puede determinar que optimizando un modelo de ML las predicciones mejoran significativamente.

3.3.5. Comparación de los valores del conjunto de datos y los predichos por el modelo optimizado.

Una vez optimizado el modelo se realizó una evaluación grafica de la predicción de las CC Y CR, donde se puede determinar que los valores predichos son cada vez más cercanos a los valores reales del conjunto de datos por lo que se pueden tomar como fiables para los diferentes estudios de análisis.

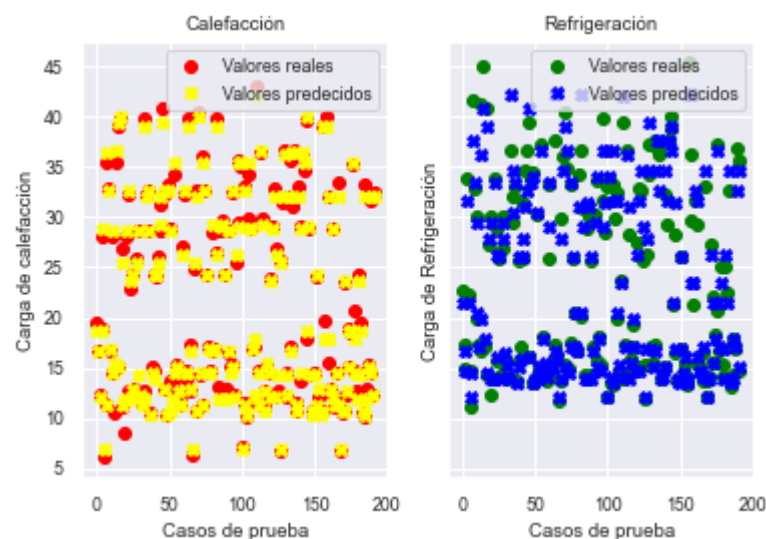


Figura 6-3: Comparación de los valores predichos del modelo experimental 2 optimizado

Realizado por: Charanchi J. 2022

3.4. Modelo experimental 3 basado en Random Forest Regression.

Como paso número uno para la construcción de modelo experimental 3 basado en Random Forest Regression se importó las librerías necesarias utilizando la biblioteca de sklearn una vez importada las librerías se procedió a la creación del modelo en cuyo caso se lo nombro reg , así creado el modelo se lo entreno para que realice las predicciones antes mencionadas.

Así también se imprimió la precisión que nos arroja el modelo en este caso siendo del 97,83%, tal y como se muestra en el siguiente código.

Random Forest Regressor

```
# Carga el modelo de random forest regressor
from sklearn.ensemble import RandomForestRegressor

# crea el modelo de random forest regressor
reg= RandomForestRegressor(n_estimators = 200, random_state = 42)
# Ajuste de los datos de (x) y (y) para la regresión
reg.fit(x_train, y_train)
y_pred3 = reg.predict(x_test)
# Predice la respuesta del modelo
acc_rfr = round(reg.score(x_test,y_test)* 100, 2)
print(" Precisión del modelo de regresión de Random Forest
:",acc_rfr,"%")
Precisión del modelo de regresión de Random Forest: 97.83 %
```

3.4.1. Métricas de evaluación del modelo experimental 3.

```
Error absoluto medio (MAE): 0.7490816406250009
Desviación: 0.9793553755100953
Error cuadrático medio (MSE): 1.9023124965169262
Raíz del Error cuadrático medio (RMSE): 1.379243450779059
R cuadrado: 0.9782649267370356
```

A continuación, se dará a conocer el resultado de las métricas de evaluación:

- El error absoluto medio (MAE): es de 0,74, es decir este valor representa la distancia a la cual se encuentra el valor de que la predicción con respecto al valor real, por lo que se puede decir que el resultado de la predicción no esta tan alejada al valor que nos da en el conjunto de datos.
- La Desviación: es de 0,97, este valor representa el error que puede tener el valor predicho con el valor que se nos dio en el conjunto de datos al estar tan cercano a 1 se puede considerar que el valor predicho si es fiable.
- El error cuadrático medio (MSE): es de 1,90, este valor representa la medida de qué tan cerca está una línea ajustada de los puntos de los datos por lo que se puede decir que varios de los

valores predichos son lejanos a los reales por lo que el modelo presentara errores leves en sus predicciones.

- La raíz del error cuadrático medio (RMSE): es de 1,37, este valor representa la distancia, en promedio, de un punto de datos de la línea ajustada, medida a lo largo de una línea vertical por lo que se puede decir que el residuo es muy grande.
- El R2 cuadrado: es de 0,97, este valor representa el porcentaje de variación que existe entre la variable de respuesta y su relación con una o más variables predictoras por lo que al estar cercana al 100% nos determina que la predicción del modelo es muy eficaz.

Así también para conocer de manera grafica la precisión de evaluación del mejor modelo obtenido en el ME3 y comprobar que tan fiables son los resultados obtenidos, se aplicó la técnica de validación cruzada (CV), para obtener una aproximación más real de la precisión del modelo y a su vez para conocer con que precisión puede generalizar nuevos datos por lo que con la ayuda de un gráfico de dispersión se puede visualizar que tan cercanos son los datos predichos a una recta normal tal y como se muestra en el siguiente código.

```
from sklearn.model_selection import cross_val_predict

# Genera estimaciones de la validación cruzada para cada punto de
datos de entrada
predicted = cross_val_predict(reg, x, y, cv=10)

# Traza las salidas previstas con respecto a las medidas
fig, ax = plt.subplots()
ax.scatter(y, predicted)
ax.plot([y.min(), y.max()], [y.min(), y.max()], 'k--', lw=4)
ax.set_xlabel('Valor medido')
ax.set_ylabel('Predicción')
plt.show()
```

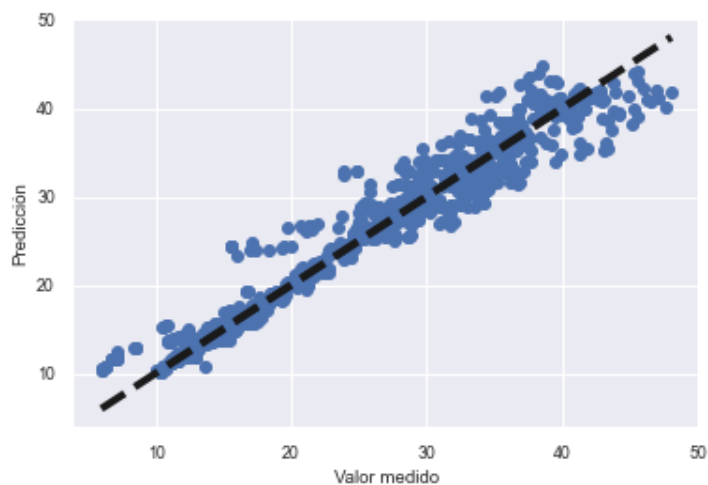


Figura 7-3: Precisión del modelo experimental 3

Realizado por: Charanchi J. 2022

3.4.2. Comparación de los valores del conjunto de datos y los predichos por el modelo.

Una vez realizada la predicción de las CC y CR se pudo determinar que los valores predichos por el modelo son muy cercanos a los valores reales del conjunto de datos por lo que se pueden tomar como fiables para los diferentes estudios de análisis.

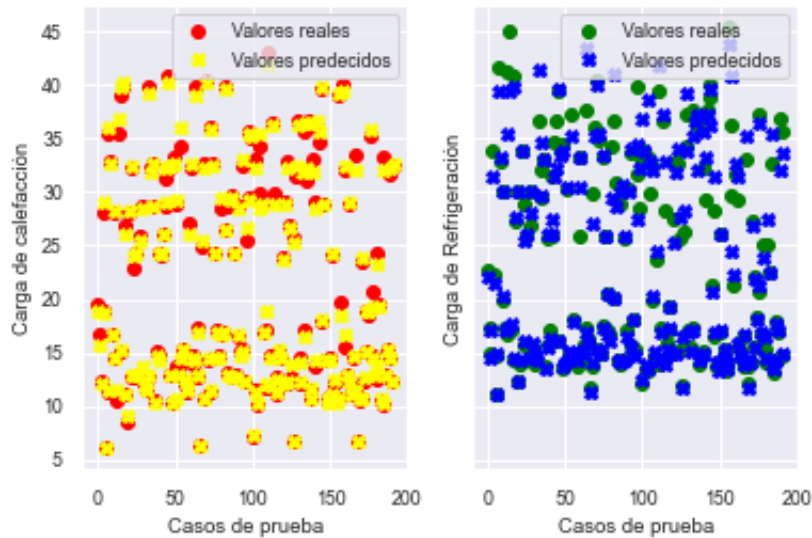


Figura 8-3: Comparación de los valores predichos del modelo experimental 3

Realizado por: Charanchi J. 2022

3.4.3. Optimización del modelo experimental 3.

Para la optimización del modelo experimental 3 se decidió encontrar los mejores parámetros para el rendimiento de este modelo, en este caso los parámetros a mejorar son la máxima profundidad que tiene el árbol, la mínima división que pueden tener las muestras, el mínimo de muestras de hojas que se pueden generar y la estimación N la cual es el número de árboles que se puede generar para que el modelo de su máximo rendimiento, tal y como se muestra en el siguiente código.

```
# Encuentra los mejores parámetros de optimización para el modelo de  
random forest regressor
```

```
f, axarr = plt.subplots(2, 2)
```

```
# Max Profundidad
```

```
rf_acc = []
```

```
rf_depth = range(1,11)
```

```
for i in rf_depth:
```

```
    rf = RandomForestRegressor(random_state=123, max_depth=i)
```

```
    rf.fit(x_train, y_train)
```

```
    rf_acc.append(rf.score(x_test, y_test))
```



```

axarr[0, 0].plot(rf_depth,rf_acc)
axarr[0, 0].set_title('Max Profundidad')

#Min División de muestras
rf_acc = []
rf_samples_split = range(10,21)
for i in rf_samples_split:
    rf = RandomForestRegressor(random_state=123, min_samples_split=i)
    rf.fit(x_train, y_train)
    rf_acc.append(rf.score(x_test, y_test))
axarr[0, 1].plot(rf_samples_split,rf_acc)
axarr[0, 1].set_title('Min División de muestras')

#Min Muestras de hojas
rf_acc = []
rf_samples_leaf = range(1,10)
for i in rf_samples_leaf:
    rf = RandomForestRegressor(random_state=123, min_samples_leaf=i)
    rf.fit(x_train, y_train)
    rf_acc.append(rf.score(x_test, y_test))

axarr[1, 0].plot(rf_samples_leaf,rf_acc)
axarr[1, 0].set_title('Min Muestras de hoja')

#Estimación de N
rf_acc = []
rf_estimators = range(50,59)
for i in rf_estimators:
    rf = RandomForestRegressor(random_state=123, n_estimators=i)
    rf.fit(x_train, y_train)
    rf_acc.append(rf.score(x_test, y_test))

axarr[1, 1].plot(rf_estimators,rf_acc)
axarr[1, 1].set_title('Estimación de N ')

plt.show()

```

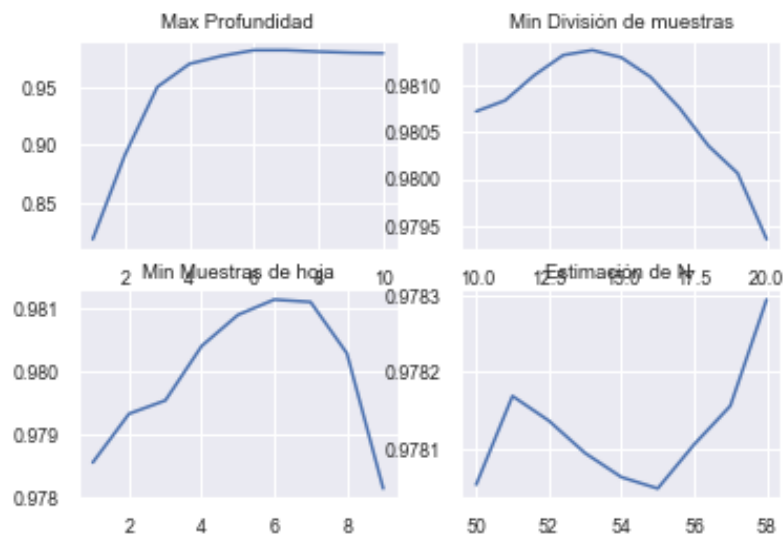


Figura 9-3: Mejores parámetros del modelo experimental 3

Realizado por: Charanchi J. 2022

Como se puede observar en la Figura 18-3 se procedió buscando los mejores parámetros en el conjunto de datos para luego decidir mediante graficas cual es el mejor valor que tomar en cuenta y predecir las variables respuestas.

3.4.4. Métricas de evaluación del modelo experimental 3 después de la optimización.

```
Error absoluto medio (MAE): 0.791222841727711
Desviación: 0.9813459327807216
Error cuadrático medio (MSE): 1.7185460223977014
Raíz del Error cuadrático medio (RMSE): 1.3109332638993112
R cuadrado: 0.9805231835361063
```

Como se muestra en el código anterior, las métricas que arroja el modelo después de optimizar los parámetros antes definidos son mejores con respecto al modelo sin optimizar por lo que se puede determinar que optimizando un modelo de ML las predicciones mejoran significativamente.

3.4.5. Comparación de los valores del conjunto de datos y los predichos por el modelo optimizado.

Una vez optimizado el modelo se realizó una evaluación grafica de la predicción de las CC Y CR, donde se puede determinar que los valores predichos son cada vez más cercanos a los valores reales del conjunto de datos por lo que se pueden tomar como fiables para los diferentes estudios de análisis.

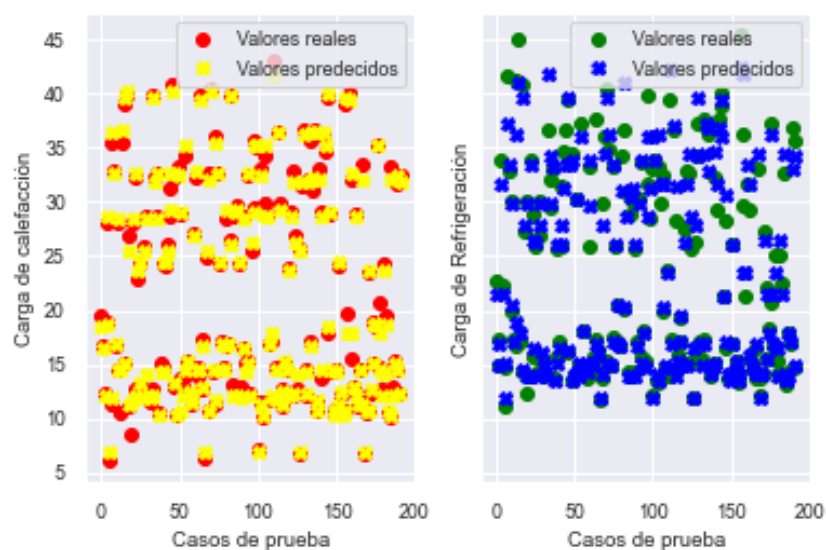


Figura 10-3: Comparación de los valores predichos del modelo experimental 3 optimizado

Realizado por: Charanchi J. 2022

3.5. Modelo experimental 4 basado en Extra Tree Regression.

Como paso número uno para la construcción de modelo experimental 4 basado en Extra Tree Regression se importó las librerías necesarias utilizando la biblioteca de sklearn una vez importada las librerías se procedió a la creación del modelo en cuyo caso se lo nombro reg , así creado el modelo se lo entreno para que realice las predicciones antes mencionadas.

Así también se imprimió la precisión que nos arroja el modelo en este caso siendo del 97,6, tal y como se muestra en el siguiente código.

Extra Trees Regressor

```
# Carga el clasificador de Extra Trees Regressor
from sklearn.ensemble import ExtraTreesRegressor

# crea el modelo de regresión
etr_model = ExtraTreesRegressor(random_state=123)
# Ajuste de los datos de (x) y (y) para la regresión
etr_model.fit(x_train, y_train)
y_pred4 = etr_model.predict(x_test)
# Predice la respuesta del modelo
acc_etr = round(etr_model.score(x_test,y_test)* 100, 2)
print(" Precisión del modelo de regresión de Extra Trees
:",acc_etr,"%")
Precisión del modelo de regresión de Extra Trees: 97.6 %
```

3.5.1. Métricas de evaluación del modelo experimental.

```
Error absoluto medio (MAE): 0.7357002604166669
Desviación: 0.9772375326944949
Error cuadrático medio (MSE): 2.096533015338534
Raíz del Error cuadrático medio (RMSE): 1.447940957131379
R cuadrado: 0.9760068668779631
```

A continuación, se dará a conocer el resultado de las métricas de evaluación:

- El error absoluto medio (MAE): es de 0,73, es decir este valor representa la distancia a la cual se encuentra el valor de que la predicción con respecto al valor real, por lo que se puede decir que el resultado de la predicción no esta tan alejada al valor que nos da en el conjunto de datos.
- La Desviación: es de 0,97, este valor representa el error que puede tener el valor predicho con el valor que se nos dio en el conjunto de datos al estar tan cercano a 1 se puede considerar que el valor predicho si es fiable.

- El error cuadrático medio (MSE): es de 2,09, este valor representa la medida de qué tan cerca está una línea ajustada de los puntos de los datos por lo que se puede decir que varios de los valores predichos son lejanos a los reales por lo que el modelo presentara errores leves en sus predicciones.
- La raíz del error cuadrático medio (RMSE): es de 1,44, este valor representa la distancia, en promedio, de un punto de datos de la línea ajustada, medida a lo largo de una línea vertical por lo que se puede decir que el residuo es muy grande.
- El R2 cuadrado: es de 0,97, este valor representa el porcentaje de variación que existe entre la variable de respuesta y su relación con una o más variables predictoras por lo que al estar cercana al 100% nos determina que la predicción del modelo es muy eficaz.

Así también para conocer de manera grafica la precisión de evaluación del mejor modelo obtenido en el ME4 y comprobar que tan fiables son los resultados obtenidos, se aplicó la técnica de validación cruzada (CV), para obtener una aproximación más real de la precisión del modelo y a su vez para conocer con que precisión puede generalizar nuevos datos por lo que con la ayuda de un gráfico de dispersión se puede visualizar que tan cercanos son los datos predichos a una recta normal tal y como se muestra en el siguiente código.

```
from sklearn.model_selection import cross_val_predict
# Genera estimaciones de la validación cruzada para cada punto de
datos de entrada
predicted = cross_val_predict(etr_model, x, y, cv=10)

# Traza las salidas previstas con respecto a las medidas
fig, ax = plt.subplots()
ax.scatter(y, predicted)
ax.plot([y.min(), y.max()], [y.min(), y.max()], 'k--', lw=4)
ax.set_xlabel('Valor medido')
ax.set_ylabel('Predicción')
plt.show()
```

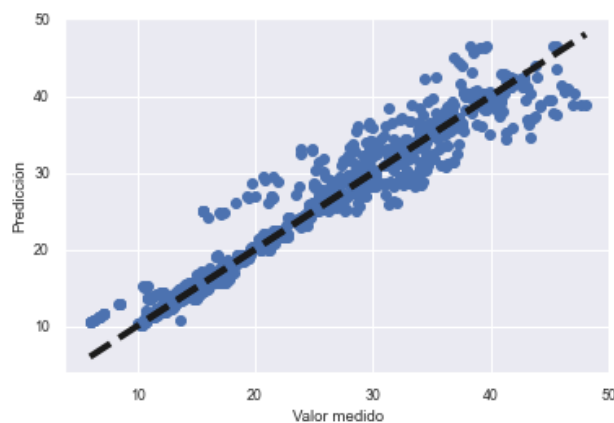


Figura 11-3: Precisión del modelo experimental 4

Realizado por: Charanchi J. 2022

3.5.2. Comparación de los valores del conjunto de datos y los predichos por el modelo.

Una vez realizada la predicción de las CC y CR se pudo determinar que los valores predichos por el modelo son muy cercanos a los valores reales del conjunto de datos por lo que se pueden tomar como fiables para los diferentes estudios de análisis.

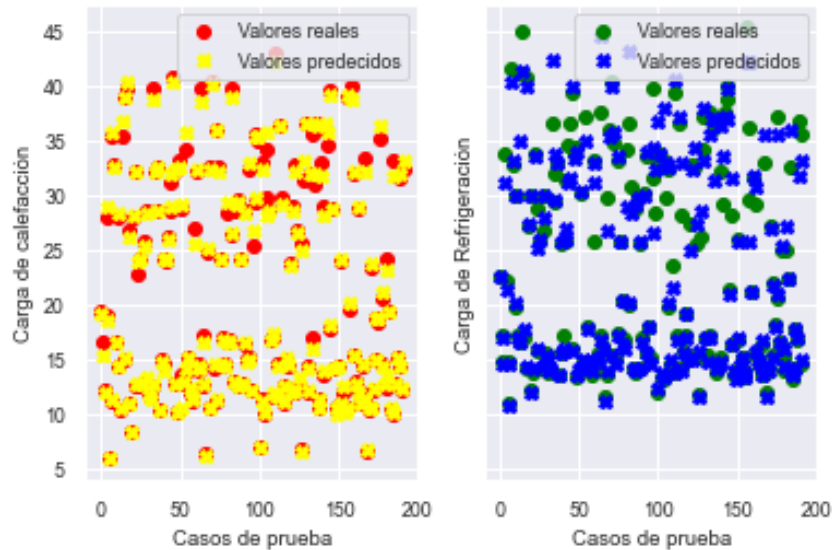


Figura 12-3: Comparación de los valores predichos del modelo experimental 4

Realizado por: Charanchi J. 2022

3.5.3. Optimización del modelo experimental 4.

Para la optimización del modelo experimental 4 se decidió encontrar los mejores parámetros para el rendimiento de este modelo, en este caso los parámetros a mejorar son la máxima profundidad que tiene el árbol, la mínima división que pueden tener las muestras, el mínimo de muestras de hojas que se pueden generar y la estimación N la cual es el número de árboles que se puede generar para que el modelo de su máximo rendimiento, tal y como se muestra en el siguiente código.

```
# Encuentra los mejores parámetros de optimización para el modelo de  
extra trees regressor
```

```
f, axarr = plt.subplots(2, 2)
```

```
# Max Profundidad
```

```
etr_acc = []
```

```
etr_depth = range(1,11)
```

```
for i in etr_depth:
```

```
    etr = ExtraTreesRegressor(random_state=123, max_depth=i)
```

```
    etr.fit(x_train, y_train)
```

```
    etr_acc.append(etr.score(x_test, y_test))
```

```

axarr[0, 0].plot(etr_depth,etr_acc)
axarr[0, 0].set_title('Max Profundidad')

#Min División de muestras
etr_acc = []
etr_samples_split = range(16,26)
for i in etr_samples_split:
    etr = ExtraTreesRegressor(random_state=123, min_samples_split=i)
    etr.fit(x_train, y_train)
    etr_acc.append(etr.score(x_test, y_test))
axarr[0, 1].plot(etr_samples_split,etr_acc)
axarr[0, 1].set_title('Min División de muestras')

#Min Muestras de hojas
etr_acc = []
etr_samples_leaf = range(3,8)
for i in etr_samples_leaf:
    etr = ExtraTreesRegressor(random_state=123, min_samples_leaf=i)
    etr.fit(x_train, y_train)
    etr_acc.append(etr.score(x_test, y_test))

axarr[1, 0].plot(etr_samples_leaf,etr_acc)
axarr[1, 0].set_title('Min Muestras de hojas')

#Estimación de N
etr_acc = []
etr_estimators = range(40,46)
for i in etr_estimators:
    etr = ExtraTreesRegressor(random_state=123, n_estimators=i)
    etr.fit(x_train, y_train)
    etr_acc.append(etr.score(x_test, y_test))

axarr[1, 1].plot(etr_estimators,etr_acc)
axarr[1, 1].set_title('Estimación de N')

plt.show()

```

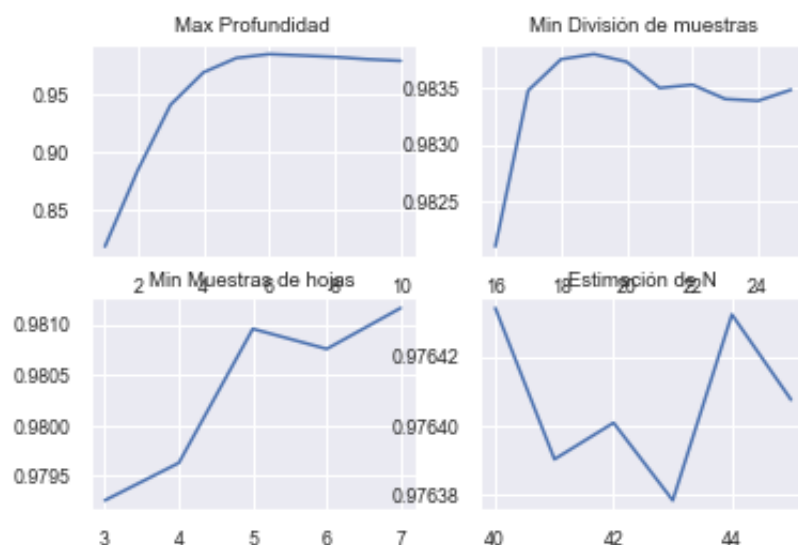


Figura 13-3: Mejores parámetros del modelo experimental 4

Realizado por: Charanchi J. 2022

Como se puede observar en la Figura 26-3 se procedió buscando los mejores parámetros en el conjunto de datos para luego decidir mediante graficas cual es el mejor valor que tomar en cuenta y predecir las variables respuestas.

3.5.4. Métricas de evaluación del modelo experimental 4 después de la optimización.

```
Error absoluto medio (MAE): 0.7628673413104428
Desviación: 0.9830792326382648
Error cuadrático medio (MSE): 1.5611992900631708
Raíz del Error cuadrático medio (RMSE): 1.2494796077020107
R cuadrado: 0.9823943335135821
```

Como se muestra en el código anterior, las métricas que arroja el modelo después de optimizar los parámetros antes definidos son mejores con respecto al modelo sin optimizar por lo que se puede determinar que optimizando un modelo de ML las predicciones mejoran significativamente.

3.5.5. Comparación de los valores del conjunto de datos y los predichos por el modelo optimizado.

Una vez optimizado el modelo se realizó una evaluación grafica de la predicción de las CC Y CR, donde se puede determinar que los valores predichos son cada vez más cercanos a los valores reales del conjunto de datos por lo que se pueden tomar como fiables para los diferentes estudios de análisis.

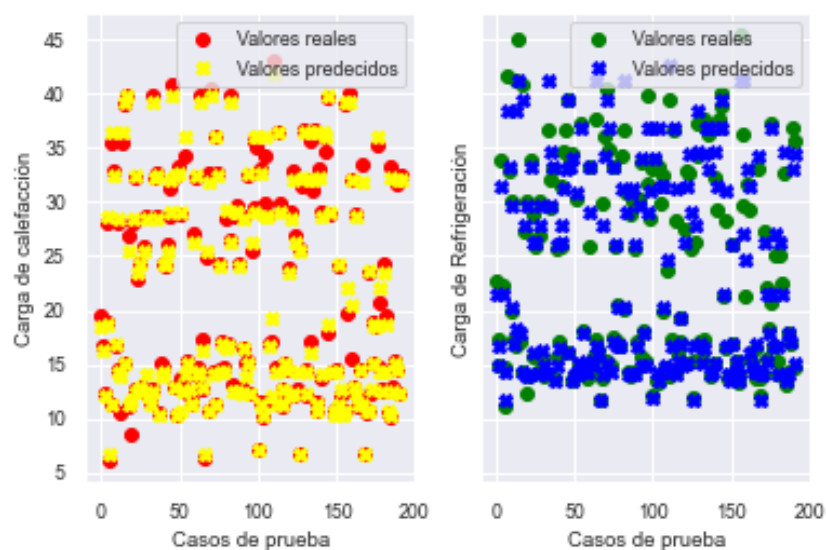


Figura 14-3: Comparación de los valores predichos del modelo experimental 4 optimizado

Realizado por: Charanchi J. 2022

3.6. Comparación de los modelos experimentales.

Como último paso del trabajo de integración curricular se evaluó la precisión de cada uno de los modelos de ML donde se puede determinar que tanto la regresión realizada por los modelos de Random Forest Regression, Decision Tree Regressor, Extra Tree Regressor y el modelo de regresión lineal son modelos aplicables a la predicción del valor de la carga de calefacción y refrigeración. La buena capacidad de predicción de los cuatro modelos es evidente por el mayor valor de las precisiones (las cuatro precisiones son >90%). Donde La regresión con el modelo de Random forest parece dar una mejor precisión que las otras tres.

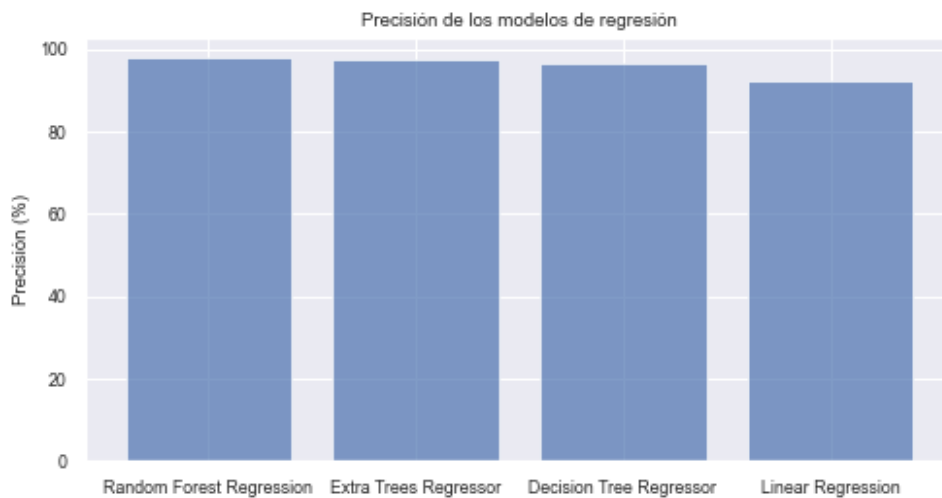


Figura 15-3: Comparación de los modelos de predicción

Realizado por: Charanchi J. 2022

CONCLUSIONES

El presente trabajo de integración curricular, se realizó con la finalidad de predecir la eficiencia energética en edificios residenciales es decir la predicción de las cargas de calefacción y refrigeración para lo cual se empleó un conjunto de datos de rendimiento energético del repositorio de la Universidad de California, Irvine (UCI), mismo datos que se utilizaron para la creación de diversos modelos de ML basados en algoritmos de regresión, los cuales fueron evaluados por diferentes métricas donde demostraron tener un buen rendimiento en la fase de entrenamiento y una buena generación de nuevos datos en la fase de prueba, lo cual se evidencio con los resultados expuestos anteriormente.

La realización del pre procesamiento del conjunto de datos de eficiencia energética que se extrajo del repositorio antes mencionado, contribuyo en gran parte al rendimiento de todos los modelos que se realizaron en esta investigación, debido a que los datos fueron analizados a través de diversos métodos exploratorios que ayudaron a determinar el comportamiento de estos, permitiéndonos detectar la existencia de errores u anomalías en la base de datos, la cuales en pasos posteriores serán corregidas; por otra parte la limpieza de los datos ayudó a mantener las variables que verdaderamente contribuyen a la predicción en base al problema que se quiere resolver, por lo que se puede decir que gracias a todos estos análisis y la optimización de parámetros en los modelos realizados se mejoró notoriamente el rendimiento de los diferentes algoritmos, ya que en las fases ultimas de evaluación se obtuvo métricas de rendimiento eficientes.

Una vez creado los diferentes modelos de aprendizaje de máquina, se pudo comprobar que las métricas de evaluación de los algoritmos superan el 90% en la precisión de los diferentes modelos destacando de todos estos el algoritmo basado en Random Forest Regressor, el cual tenía como principal métrica de evaluación el R2, mostró un porcentaje elevado con respecto al resto de modelos al momento de la predicción de las cargas de calefacción y refrigeración.

RECOMENDACIONES

Se recomienda en una investigación futura, realizar modelos de aprendizaje de máquina basados en clasificación y con un análisis de aprendizaje profundo con el fin de analizar el segundo enfoque de investigación del conjunto de datos, el cual consiste en determinar la calidad del consumo energético de acuerdo con la distribución de los parámetros de un edificio.

Se recomienda un mayor análisis de las variables del conjunto de datos, con la finalidad de evaluar si se elimina una o dos variables con poca correlación con las variables a predecir mejorando el rendimiento del modelo y en consecuencia apreciando un aumento en los porcentajes de las métricas de evaluación obtenidas.

Se recomienda realizar otros modelos basados en regresión como lo son los modelos de XGBoosts, Support Vector Regression y algoritmos basados en redes neuronales, con la finalidad de determinar si se logra mejores resultados de la predicción de las variables, ya que en la presente investigación estos algoritmos no fueron tomados en cuenta para la comparación con el modelo de regresión lineal.

Se recomienda realizar más investigaciones sobre la eficiencia energética en edificios residenciales, debido a que en el país no existen investigaciones ni normativas que hablen de este tema, a la vez que se da la sugerencia de crear una base de datos con los consumos energéticos de los edificios en el Ecuador, así como la distribución de estos y los materiales utilizados en su construcción, ya que esto ayuda a mejorar el consumo de energía en el país y crear conciencia en las personas a la hora construir estas infraestructuras, debido a que estos futuros edificios estarán basados en la sostenibilidad del planeta y el confort térmico.

BIBLIOGRAFIA

ABDELAZIZ, A., SANTOS, V. y DIAS, M.S. Machine learning techniques in the energy consumption of buildings: A systematic literature review using text mining and bibliometric analysis. *Energies*, 2021, vol. 14, ISSN 19961073. DOI 10.3390/en14227810.

ANACONDA SOFTWARE. Anaconda Individual Edition. [en línea], 2020, [Consulta: 17 febrero 2022]. Disponible en: <https://www.anaconda.com/products/individual>.

ARIF, T.M. *Introduction to Deep Learning for Engineers* [en línea], 2020,. Disponible en: <http://store.morganclaypool.com>.

ATTIA, S., ELEFThERIOU, P., XENI, F., MORLOT, R., MÉNÉZO, C., KOSTOPOULOS, V., BETSI, M., KALAITZOGLOU, I., PAGLIANO, L., CELLURA, M., ALMEIDA, M., FERREIRA, M., BARACU, T., BADESCU, V., CRUTESCU, R. y HIDALGO-BETANZOS, J.M. Overview and future challenges of nearly zero energy buildings (nZEB) design in Southern Europe. *Energy and Buildings* [en línea], vol. 155, 2017, pp. 439-458. ISSN 03787788. DOI 10.1016/j.enbuild.2017.09.043. Disponible en: <https://doi.org/10.1016/j.enbuild.2017.09.043>.

BATTA, M. Machine Learning Algorithms - A Review. *International Journal of Science and Research (IJ)* [en línea], vol. 9, 2020, pp. 381-382. DOI 10.21275/ART20203995. Disponible en: https://www.researchgate.net/profile/BattaMahesh/publication/344717762_Machine_Learning_Algorithms_A_Review/links/5f8b2365299bf1b53e2d243a/Machine-Learning_Algorithms-A-Review.pdf.

BIENVENIDO-HUERTAS, D. y RUBIO-BELLIDO, C. *Adaptive Thermal Comfort of Indoor Environment for Residential Buildings* [en línea], 2021, S.l.: s.n. ISBN 978-981-16-0905-3. Disponible en: <https://link.springer.com/10.1007/978-981-16-0906-0>.

CARDOSO, D. y FERREIRA, L. Application of predictive maintenance concepts using artificial intelligence tools. *Applied Sciences (Switzerland)*, vol. 11, 2021, pp. 1-18. ISSN 20763417. DOI 10.3390/app11010018.

DUARTE, G.R., FONSECA, L.G. da, GOLIATT, P.V.Z.C. y LEMONGE, A.C. de C. Uma comparação de técnicas de aprendizado de máquina para a previsão de cargas energéticas em

edificios. *Ambiente Construído*, vol. 17, 2017, pp. 103-115. ISSN 1678-8621. DOI 10.1590/s1678-86212017000300165.

HOPE, T.M.H. Linear regression. *Machine Learning: Methods and Applications to Brain Disorders*, 2019, pp. 67-81. DOI 10.1016/B978-0-12-815739-8.00004-3.

IRVINE, U. de C. UCI Machine Learning Repository Fertility Data Set. *Energy efficiency Data Set* [en línea], 2012 [Consulta: 17 febrero 2022]. Disponible en: <https://archive.ics.uci.edu/ml/datasets/energy+efficiency>.

KHAN, P.W., KIM, Y., BYUN, Y.C. y LEE, S.J. Influencing factors evaluation of machine learning-based energy consumption prediction. *Energies*, vol. 14, 2021, ISSN 19961073. DOI 10.3390/en14217167.

LIN, W.C., CHEN, J.S., CHIANG, M.F. y HRIBAR, M.R. Applications of artificial intelligence to electronic health record data in ophthalmology. *Translational Vision Science and Technology*, vol. 9, 2020, ISSN 21642591. DOI 10.1167/tvst.9.2.13.

MANRIQUE, E. Machine Learning: análisis de lenguajes de programación y herramientas para desarrollo. *Revista Ibérica de Sistemas e Tecnologias de Informação* [en línea], 2020, pp.586-599. Disponible en: <https://search.proquest.com/openview/c7e24c997199215aa26a39107dd2fe98/1?pqorigsite=scholar&cbl=1006393>.

MCARTHUR, J.J., SHAHBAZI, N., FOK, R., RAGHUBAR, C., BORTOLUZZI, B. y AN, A. Machine learning and BIM visualization for maintenance issue classification and enhanced data collection. *Advanced Engineering Informatics* [en línea], vol. 38, 2018, pp. 101-112. ISSN 14740346. DOI 10.1016/j.aei.2018.06.007. Disponible en: <https://www.sciencedirect.com/science/article/abs/pii/S1474034617305049?via%3Dihub>.

MEJÍAL, J. *Análisis Multivarian te con Enfoque Depend ien te EN LAS CIENCIAS DE LA ADMINISTRACIÓN COMO BASE PARA LA INNOVACIÓN*. Guadalajara, 2019, ISBN 9786079878238.

MÜLLER, A.C. y GUIDO, S., *Introduction to with Python Learning Machine*. S.I, 2017, ISBN 9781449369415.

NEBOT, À. y MUGICA, F., Energy performance forecasting of residential buildings using fuzzy approaches. *Applied Sciences (Switzerland)*, vol. 10, 2020, ISSN 20763417. DOI 10.3390/app10020720.

PHAM, A.D., NGO, N.T., HA TRUONG, T.T., HUYNH, N.T. y TRUONG, N.S., Predicting energy consumption in multiple buildings using machine learning for improving energy efficiency and sustainability. *Journal of Cleaner Production* [en línea], vol. 260, 2020, pp. 121082. ISSN 09596526. DOI 10.1016/j.jclepro.2020.121082. Disponible en: <https://www.sciencedirect.com/science/article/abs/pii/S1364032116306608>.

PYTHON SOFTWARE FOUNDATION. What is Python? Executive Summary | Python.org. [en línea], 2021 [Consulta: 23 noviembre 2021]. Disponible en: <https://www.python.org/doc/essays/blurb/>.

QIU, X., ZHANG, L., NAGARATNAM SUGANTHAN, P. y AMARATUNGA, G.A.J., Oblique random forest ensemble via Least Square Estimation for time series forecasting. *Information Sciences* [en línea], vol. 420, 2017, pp. 249-262. ISSN 00200255. DOI 10.1016/j.ins.2017.08.060. Disponible en: <http://dx.doi.org/10.1016/j.ins.2017.08.060>.

QUORA. ¿Qué es la regresión lineal? Explique con un ejemplo. - Quora. *¿Qué es la regresión lineal? Explique con un ejemplo. - Quora* [en línea], 2017, [Consulta: 6 febrero 2022]. Disponible en: <https://www.quora.com/What-is-linear-regression-Explain-with-an-example>.

RICS. Energy efficiency and residential values : a changing European landscape. *RICS insight paper* [en línea], 2019, pp. 35. Disponible en: <https://www.rics.org/globalassets/rics-website/media/knowledge/research/insights/energy-efficiency-and-residential-values.pdf%0Afile:///C:/Users/vital/Dropbox/PhD - reading list/Read-Mendeley/Market value/energy-efficiency-and-residential-values.pdf>.

ROLON-MÉRETTE, D., ROSS, M., ROLON-MÉRETTE, T. y CHURCH, K. Introduction to Anaconda and Python: Installation and setup. *The Quantitative Methods for Psychology* [en línea], vol. 16, 2020, pp. S3-S11. ISSN 2292-1354. DOI 10.20982/tqmp.16.5.s003. Disponible en: <https://www.tqmp.org/SpecialIssues/vol16-5/S003/>.

ROUHIAINEN, L. Inteligencia artificial 101 cosas que debes saber. *Alienta editorial* [en línea], 2018, pp.352. Disponible en: https://planetadelibrosar0.cdnstatics.com/libros_contenido_extra/40/39307_Inteligencia_artif

icial.pdf.

SADEGHI, A., SINAKI, R.Y., YOUNG, W.A. y WECKMAN, G.R. An intelligent model to predict energy performances of residential buildings based on deep neural networks. *Energies*, vol. 13, 2020, no. 3. ISSN 19961073. DOI 10.3390/en13030571.

SAJJAD, M., KHAN, S.U., KHAN, N., HAQ, I.U., ULLAH, A., LEE, M.Y. y BAIK, S.W. Towards efficient building designing: Heating and cooling load prediction via multi-output model. *Sensors (Switzerland)*, vol. 20, 2020, no. 22, pp. 1-19. ISSN 14248220. DOI 10.3390/s20226419.

SCHRIDER, D.R. y KERN, A.D. Supervised Machine Learning for Population Genetics: A New Paradigm. *Trends in Genetics* [en línea], vol. 34, 2018, no. 4, pp. 301-312. ISSN 13624555. DOI 10.1016/j.tig.2017.12.005. Disponible en: <https://www.sciencedirect.com/science/article/pii/S0168952517302251>.

SEKHAR ROY, S., ROY, R. y BALAS, V.E. Estimating heating load in buildings using multivariate adaptive regression splines, extreme learning machine, a hybrid model of MARS and ELM. *Renewable and Sustainable Energy Reviews*, vol. 82, 2018, pp. 4256-4268. ISSN 18790690. DOI 10.1016/j.rser.2017.05.249.

SHOBHA, G. y RANGASWAMY, S. Machine Learning. *Handbook of Statistics* [en línea]. 1. S.l.: Elsevier B.V., 2018, pp. 197-228. ISBN 9780444640420. Disponible en: <http://dx.doi.org/10.1016/bs.host.2018.07.004>.

SOLUCIONES, E. 5 de marzo: Día Mundial de la Eficiencia Energética. [en línea], 2021 [Consulta: 25 enero 2022]. Disponible en: <https://www.e4e-soluciones.com/blog-eficiencia-energetica/dia-mundial-de-la-eficiencia-energetica>.

TRISTANCHO CARVAJAL, M. *Predicción De La Demanda Energética Basada En El Confort Adaptativo Aplicado Al Sector Terciario Considerando El Cambio Climático* [en línea], 2019, S.l.: s.n. Disponible en: <https://idus.us.es/bitstream/handle/11441/90205/aomaster266.pdf?sequence=1&isAllowed=y>.

ÜNLÜ, R. y NAMLI, E. Machine learning and classical forecasting methods based decision support systems for covid-19. *Computers, Materials and Continua*, vol. 64, 2020, pp. 1383-1399. ISSN 15462226. DOI 10.32604/cmc.2020.011335.

VILLENA, F., GARCÍA, T., BALLESTEROS, P. y PELLICER, E. International Congress on Project Management and Engineering Málaga, 10. [en línea], vol. 029, 2019, pp. 10. Disponible en: http://dspace.aepro.com/xmlui/bitstream/handle/123456789/2408/AT10-001_2019.pdf?sequence=1&isAllowed=y.

WANG, Z., WANG, Y., ZENG, R., SRINIVASAN, R.S. y AHRENTZEN, S. Random Forest based hourly building energy prediction. *Energy and Buildings* [en línea], vol. 171, 2018, pp. 11-25. ISSN 03787788. DOI 10.1016/j.enbuild.2018.04.008. Disponible en: <https://doi.org/10.1016/j.enbuild.2018.04.008>.

WARREN-MYERS, G., KAIN, C. y DAVIDSON, K. The wandering energy stars: The challenges of valuing energy efficiency in Australian housing. *Energy Research and Social Science* [en línea], vol. 67, 2020, pp. 101505. ISSN 22146296. DOI 10.1016/j.erss.2020.101505. Disponible en: <https://doi.org/10.1016/j.erss.2020.101505>.

ZHANG, X.-D. Chapter 6 Machine Learning. *A Matrix Algebra Approach to Artificial Intelligence* [en línea], 2017, pp. 40-48. ISBN 026201243X. Disponible en: <https://books.google.ca/books?id=EoYBngEACAAJ&dq=mitchell+machine+learning+1997&hl=en&sa=X&ved=0ahUKEwiodmqfj8TkAhWGslkKHRCbAtoQ6AEIKjAA>.

ANEXOS

ANEXO A: PROGRAMACIÓN FINALIZADA



Resumen

El presente trabajo de integración curricular tiene como objetivo principal aplicar técnicas de análisis de aprendizaje automático (ML) basado en métodos de regresión lineal para predecir las cargas de calefacción y refrigeración en edificios residenciales, a través de la generación de ecuaciones de regresión y código en lenguaje de Python para las técnicas de ML.

Para la creación del algoritmo de aprendizaje se utilizó un conjunto de datos de rendimiento energético cuya base de datos fue extraída del Machine Learning Repository de la Universidad de California, Irvine. El conjunto de datos contiene ocho atributos (o características, denotados por $X_1 \dots X_8$) y dos respuestas (o resultados, denotados por Y_1 y Y_2). El objetivo es utilizar las ocho funciones para predecir cada una de las dos respuestas.

La primera parte está dedicada al análisis exploratorio de datos y visualizaciones, así como a la aplicación de técnicas estadísticas y análisis de componentes principales en el conjunto de datos.

En la segunda parte, desarrollamos varios métodos de clasificación directa e indirecta comparando rendimientos, con el objetivo de encontrar el modelo óptimo para predecir la eficiencia energética de los edificios.

Por último, implementamos los algoritmos de regresión lineal y regresión lineal múltiple para las predicciones y analizamos los resultados. El conjunto de datos contiene ocho atributos (o características, denotados por $X_1 \dots X_8$) y dos respuestas (o resultados, denotados por Y_1 y Y_2). El objetivo es utilizar las ocho funciones para predecir cada una de las dos respuestas.

Específicamente:

Las variables de entrada son:

- X_1 Compacidad relativa
- X_2 Superficie - m^2
- X_3 Superficie de la pared - m^2
- X_4 Superficie del techo - m^2
- X_5 Altura total - m
- X_6 Orientación - 2: Norte, 3: Este, 4: Sur, 5: Oeste
- X_7 Área de acristalamiento: 0%, 10%, 25%, 40% (del área del piso)
- X_8 Distribución del área de acristalamiento (variación) - 1: uniforme, 2: norte, 3: este, 4: sur, 5: oeste

Las variables de salida son:

- Y_1 Carga de calefacción - kWh
- Y_2 Carga de refrigeración - kWh

Introducción

En la actualidad el calentamiento global se ha convertido en el principal problema de la humanidad, puesto que los cambios de temperatura que afectan a el planeta han provocado que las personas se ayuden de diversos sistemas electrónicos para controlar de la calefacción y la refrigeración dentro de sus hogares, creando así un consumo excesivo de energía ya que los edificios residenciales no cuenta con entornos adecuados para el uso de estos aparatos, por lo que el rendimiento energético es un parámetro importante a tener en cuenta al momento de construir un edificio. Dependiendo las condiciones climáticas del entorno y de las características que se le dé a la construcción, un edificio puede guardar alrededor del 40% del uso total de la energía. por lo que predecir las cargas de calefacción y refrigeración de un edificio es muy importante, debido a esto se puede decir que la mejor manera para mejorar la eficiencia energética es encontrar soluciones óptimas entre los diferentes diseños en la fase inicial, así como en la fase de funcionamiento después de que el edificio haya sido terminado.

De igual manera se puede decir que para garantizar un consumo de energía sostenible se debe realizar un examen adecuado del rendimiento energético de los edificios (REE) y un diseño óptimo del sistema de calefacción, ventilación y aire acondicionado (CVAC). Aunque en muchos lugares alrededor del mundo han controlado estas medidas, todavía existe un alto consumo de energía y se prevé que aumente a un más. De acuerdo con los elementos mencionados anteriormente, en este estudio pretende aplicar técnicas de aprendizaje automático basado en regresión lineal, analizando datos reales de edificios residenciales, extraídos de una base de datos del Machine Learning Repository de la Universidad de California, Irvine. los cuales permiten obtener predicciones mas cercanas del consumo de energía en diversos entornos, desarrollando así una herramienta de evaluación que ayude a optimizar la eficiencia energética.

Importación de librerías

```
In [1]: #Importación de todas Las bibliotecas
import numpy as np
import pandas as pd
from sklearn.preprocessing import StandardScaler

#construcción del modelo
import statsmodels.api as sm
import statsmodels.formula.api as smf

#evaluación del modelo
from sklearn.feature_selection import RFE
from statsmodels.stats.outliers_influence import variance_inflation_factor
from sklearn import metrics
from sklearn.metrics import precision_recall_curve
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import explained_variance_score
from sklearn.metrics import mean_squared_error
from sklearn.metrics import r2_score

#validación del modelo
from sklearn.model_selection import train_test_split

#visualización
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

# Supresión de advertencias
import warnings
warnings.filterwarnings('ignore')

# Personalización de la visualización de datos
pd.set_option('display.max_columns', 100)
```

Importación & lectura del conjunto de datos

Los datos para la creación del algoritmo de aprendizaje para el análisis de eficiencia energética en edificios residenciales basado en regresión lineal se lo va ha realizar a través de una base de datos de Machine Learning Repository de la Universidad de California, Irvine.

Enlace de descarga del dataset

<https://archive.ics.uci.edu/ml/datasets/energy+efficiency>

Primero se carga los datos que están en formato CSV, con ayuda de la librería pandas

- Se usa `pd.read_csv` debido a que importamos a pandas como `pd`

Descripción del conjunto de datos:

Se tiene un análisis de energía utilizando 12 formas de edificios diferentes simuladas en Ecotect. Los edificios se diferencian en cuanto a la superficie acristalada, la distribución de la superficie acristalada y la orientación, entre otros parámetros. Simulamos varios escenarios en función de las características antes mencionadas para obtener 768 formas de edificios.

El conjunto de datos comprende 768 muestras y 8 características, con el objetivo de predecir dos respuestas de valor real.

Explicación de las variables dentro del conjunto de datos:

Para mayor comodidad, los significados variables se dan a continuación.

Compacidad relativa	Superficie	Superficie de la pared	Superficie del techo	Altura total		
Superficie/volumen	m ²	m ²	m ²	m		
	Valor	2	3	4	5	
	Orientación	Norte	Este	Sur	Oeste	
	Valor		0	.10	.25	.40
	Acrilamiento % de la superficie del suelo	0	10%	25%	40%	
Valor	0	1	2	3	4	5
Distribución del área de acristamiento	Ninguno	Uniforme	Sesgado al Norte	Sesgado al Este	Sesgado al Sur	Sesgado al Oeste
	Carga de calefacción		Carga de refrigeración			
	kWh		kWh			

Todos los edificios tienen el mismo volumen, con diferentes superficies y dimensiones individuales. Los materiales de construcción para las paredes, pisos, techos y ventanas fueron modelados como iguales, y como tales tienen los mismos valores aislantes.

```
In [2]: # Lectura del conjunto de datos de EFICIENCIA ENERGETICA
datos= pd.read_csv("C:/Users/USUARIO/Pictures/EnergyEfficiency/ENB2012_data.csv")

# Renombramos Las columnas
datos.columns = ['Compacidad_relativa', 'Superficie', 'Superficie_de_la_pared', 'Superficie_del_techo', 'Altura_total',
                 'Orientación', 'Área_de_acristalamiento', 'Distribución_del_área_de_acristalamiento', 'Carga_de_calentamiento', 'Carga_de_refrigeración']

# Impresión de Los datos
datos
```

Out[2]:

	Compacidad_relativa	Superficie	Superficie_de_la_pared	Superficie_del_techo	Altura_total	Orientación	Área_de_acristalamiento	Distribución_del_área_de_acristalamiento
0	0.98	514.5	294.0	110.25	7.0	2	0.0	
1	0.98	514.5	294.0	110.25	7.0	3	0.0	
2	0.98	514.5	294.0	110.25	7.0	4	0.0	
3	0.98	514.5	294.0	110.25	7.0	5	0.0	
4	0.90	583.5	318.5	122.50	7.0	2	0.0	
...
763	0.64	784.0	343.0	220.50	3.5	5	0.4	
764	0.62	808.5	367.5	220.50	3.5	2	0.4	
765	0.62	808.5	367.5	220.50	3.5	3	0.4	
766	0.62	808.5	367.5	220.50	3.5	4	0.4	
767	0.62	808.5	367.5	220.50	3.5	5	0.4	

768 rows x 10 columns

```
In [3]: # Impresión de La forma del conjunto de datos
f,c = datos.shape
print(f"Forma del conjunto de datos de eficiencia energetica: {datos.shape}")
print(f"Numero de filas: {f}")
print(f"Numero de columnas: {c}")
```

Forma del conjunto de datos de eficiencia energetica: (768, 10)
Numero de filas: 768
Numero de columnas: 10

```
In [4]: # Descripción estadística de Los datos
datos.describe().transpose()
```

Out[4]:

	count	mean	std	min	25%	50%	75%	max
Compacidad_relativa	768.0	0.764167	0.105777	0.62	0.6825	0.75	0.8300	0.98
Superficie	768.0	671.708333	88.086116	514.50	808.3750	673.75	741.1250	808.50
Superficie_de_la_pared	768.0	318.500000	43.828481	245.00	294.0000	318.50	343.0000	416.50
Superficie_del_techo	768.0	176.604167	45.165950	110.25	140.8750	183.75	220.5000	220.50
Altura_total	768.0	5.250000	1.751140	3.50	3.5000	5.25	7.0000	7.00
Orientación	768.0	3.500000	1.118783	2.00	2.7500	3.50	4.2500	5.00
Área_de_acristalamiento	768.0	0.234375	0.133221	0.00	0.1000	0.25	0.4000	0.40
Distribución_del_área_de_acristalamiento	768.0	2.812500	1.550980	0.00	1.7500	3.00	4.0000	5.00
Carga_de_calentamiento	768.0	22.307201	10.090196	8.01	12.9925	18.95	31.8675	43.10
Carga_de_refrigeración	768.0	24.587760	9.513306	10.90	15.6200	22.08	33.1325	48.03

```
In [7]: # Imprimimos La información de Los datos que contiene del conjunto de datos
datos.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 10 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   Compacidad_relativa                   768 non-null   float64
1   Superficie                             768 non-null   float64
2   Superficie_de_la_pared                 768 non-null   float64
3   Superficie_del_techo                   768 non-null   float64
4   Altura_total                           768 non-null   float64
5   Orientación                            768 non-null   int64
6   Área_de_acristalamiento                768 non-null   float64
7   Distribución_del_área_de_acristalamiento 768 non-null   int64
8   Carga_de_calentamiento                  768 non-null   float64
9   Carga_de_refrigeración                  768 non-null   float64
dtypes: float64(8), int64(2)
memory usage: 60.1 KB
```

Comprobación de valores faltantes

```
In [5]: # comprobación de valores faltantes
datos.isna().sum()
```

```
Out[5]: Compacidad_relativa      0
Superficie      0
Superficie_de_la_pared      0
Superficie_del_techo      0
Altura_total      0
Orientación      0
Área_de_acristalamiento      0
Distribución_del_área_de_acristalamiento      0
Carga_de calefacción      0
Carga_de_refrigeración      0
dtype: int64
```

El conjunto de datos no contiene valores desconocidos

```
In [6]: # comprobamos Los nombres de Las columnas
datos.columns
```

```
Out[6]: Index(['Compacidad_relativa', 'Superficie', 'Superficie_de_la_pared',
'Superficie_del_techo', 'Altura_total', 'Orientación',
'Área_de_acristalamiento', 'Distribución_del_área_de_acristalamiento',
'Carga_de_calefacción', 'Carga_de_refrigeración'],
dtype='object')
```

Cada característica tiene una escala diferente, ya que podemos ver los valores mínimos y máximos para cada una de las variables. Para obtener una mejor escala, es bueno normalizar los datos porque hace que las distribuciones sean mejores.

Exploración de los datos del conjunto de datos

```
In [8]: ax = plt.figure(figsize=(12,12)).gca()
datos.hist(ax=ax)
```

```
Out[8]: array([[<AxesSubplot:title={'center':'Compacidad_relativa'}>,
<AxesSubplot:title={'center':'Superficie'}>,
<AxesSubplot:title={'center':'Superficie_de_la_pared'}>],
[<AxesSubplot:title={'center':'Superficie_del_techo'}>,
<AxesSubplot:title={'center':'Altura_total'}>,
<AxesSubplot:title={'center':'Orientación'}>],
[<AxesSubplot:title={'center':'Área_de_acristalamiento'}>,
<AxesSubplot:title={'center':'Distribución_del_área_de_acristalamiento'}>,
<AxesSubplot:title={'center':'Carga_de_calefacción'}>],
[<AxesSubplot:title={'center':'Carga_de_refrigeración'}>,
<AxesSubplot:;>, <AxesSubplot:;>], dtype=object)
```

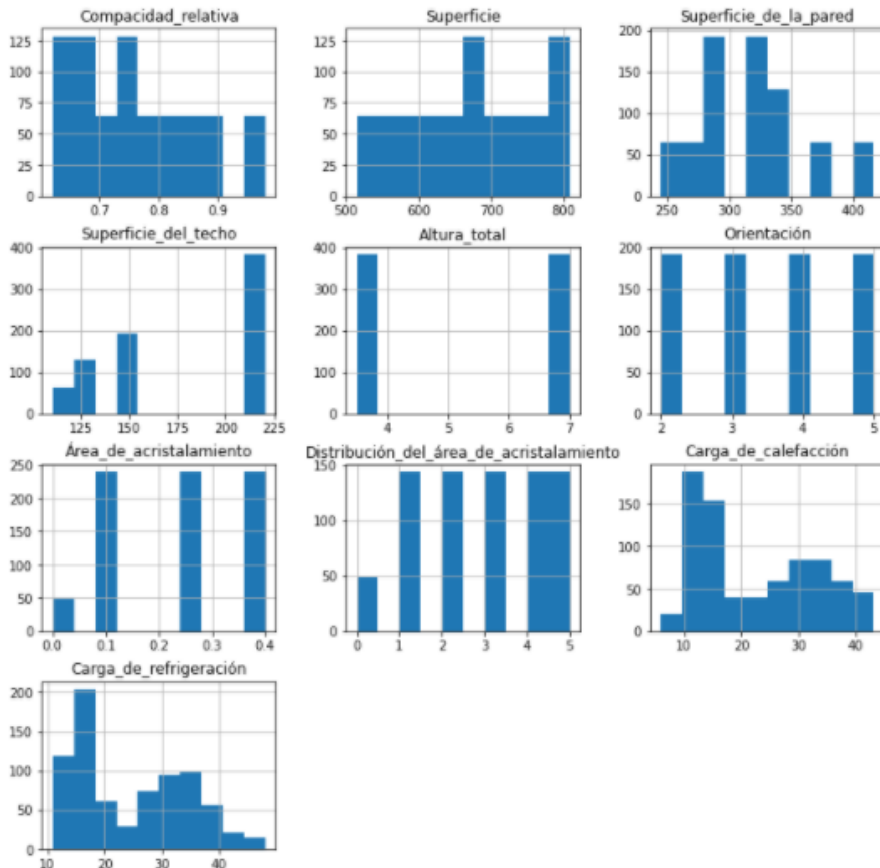


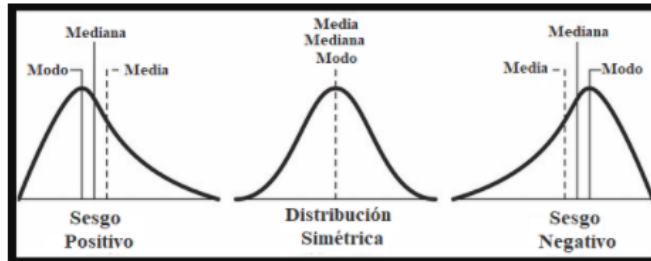
Diagrama de estimación de densidad del núcleo:

En estadística, la estimación de la densidad del núcleo (KDE) es una forma no paramétrica de estimar la función de densidad de probabilidad de una variable aleatoria. La estimación de la densidad del núcleo es un problema fundamental de suavizado de datos donde se hacen inferencias sobre la población, basadas en una muestra de datos finitos.

Distorsión

Es el grado de distorsión de la curva de campana simétrica o la distribución normal. Mide la falta de simetría en la distribución de datos. Diferencia los valores extremos en una cola frente a la otra. Una distribución simétrica tendrá una asimetría de 0.

Hay dos tipos de asimetría: positiva y negativa

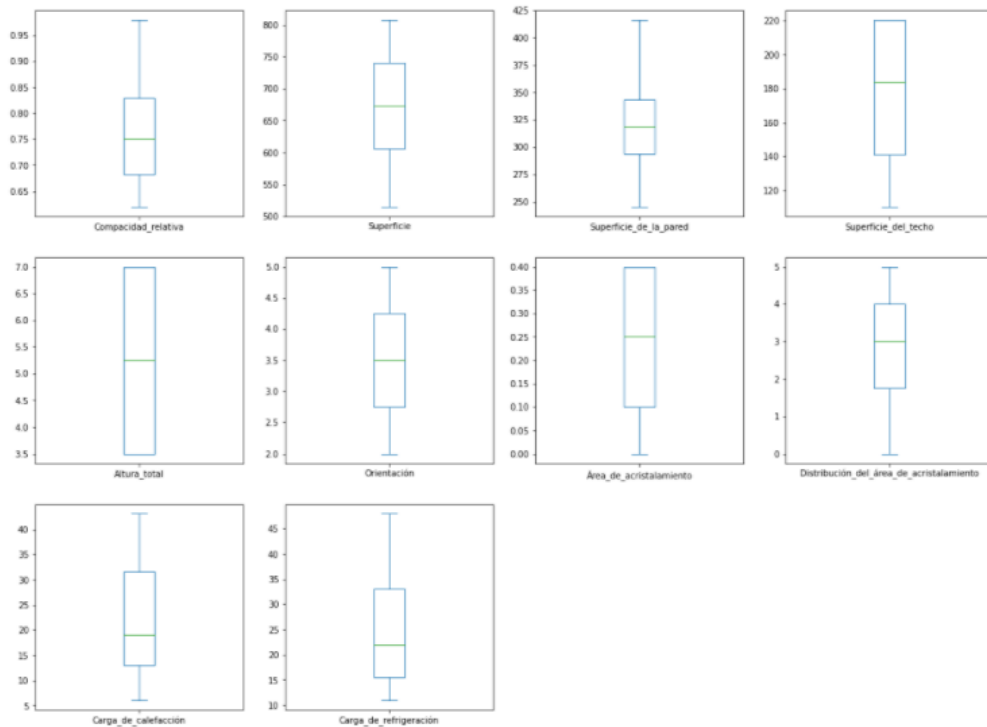


La asimetría positiva significa cuando la cola en el lado derecho de la distribución es más larga o más gorda. La media y la mediana serán mayores que el modo.

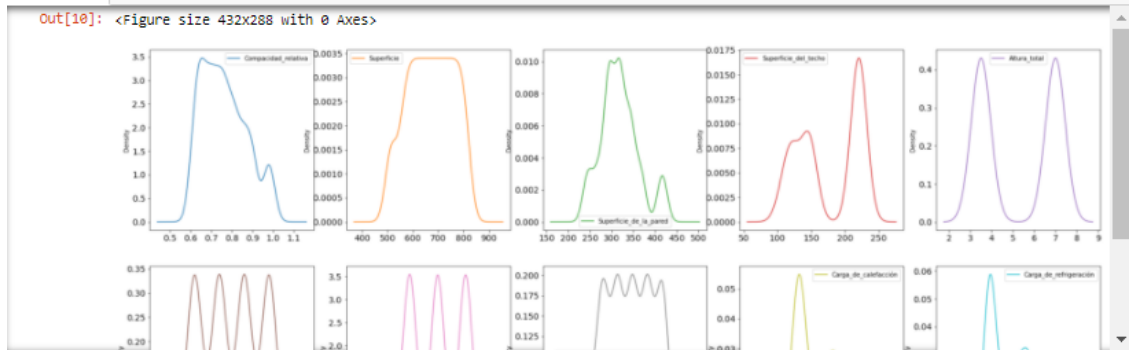
La asimetría negativa es cuando la cola del lado izquierdo de la distribución es más larga o más gorda que la cola del lado derecho. La media y la mediana serán menores que el modo.

```
In [9]: # Trazado del cuadro para todas Las variables continuas
datos.plot(kind='box', subplots=True, layout=(3,4), sharex=False, sharey=False, figsize=(20,15))
```

```
Out[9]: Compacidad_relativa      AxesSubplot(0.125,0.657941;0.168478x0.222059)
Superficie      AxesSubplot(0.327174,0.657941;0.168478x0.222059)
Superficie_de_la_pared      AxesSubplot(0.529348,0.657941;0.168478x0.222059)
Superficie_del_techo      AxesSubplot(0.731522,0.657941;0.168478x0.222059)
Altura_total      AxesSubplot(0.125,0.391471;0.168478x0.222059)
Orientación      AxesSubplot(0.327174,0.391471;0.168478x0.222059)
Área_de_acristalamiento      AxesSubplot(0.529348,0.391471;0.168478x0.222059)
Distribución_del_área_de_acristalamiento      AxesSubplot(0.731522,0.391471;0.168478x0.222059)
Carga_de calefacción      AxesSubplot(0.125,0.125;0.168478x0.222059)
Carga_de_refrigeración      AxesSubplot(0.327174,0.125;0.168478x0.222059)
dtype: object
```



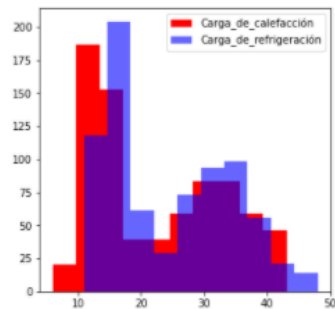
```
In [10]: # Comprueba la densidad de cada variable con KDE plot
datos.plot(kind='kde', subplots=True, layout=(2,5), figsize=(28,12), fontsize=14, sharex=False, sharey=False)
plt.figure()
```



En la gráfica anterior, la mayoría de las características tienen un sesgo positivo, excepto algunas y la "orientación" y la "altura_total" tienen una distribución bastante simétrica.

```
In [11]: # Visualización de La distribución de Los valores de Las variables de salida
plt.figure(figsize=(5,5))
plt.hist((datos.Carga_de_calefacción), alpha=1, color = 'r')
plt.hist((datos.Carga_de_refrigeración), alpha=0.6, color = 'b')
plt.legend(['Carga_de_calefacción', 'Carga_de_refrigeración'])
```

Out[11]: <matplotlib.legend.Legend at 0x1ca42fab400>



Distribución gaussiana (también conocida como distribución normal)

La distribución gaussiana o normal es una función de probabilidad que describe cómo se distribuyen los valores de una variable.

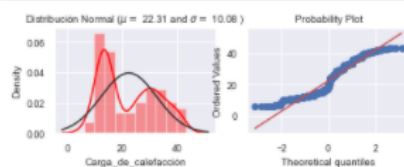
Es una curva en forma de campana, y se supone que durante cualquier medición los valores seguirán una distribución normal con un número igual de mediciones por encima y por debajo del valor medio.

```
In [12]: # Comprueba la distribución normal de La carga de calefacción con un histograma y máxima probabilidad gaussiana
from scipy import stats
from scipy.stats import norm
sns.set(font_scale=.8)

ax1 = plt.subplot(221)
sns.distplot(datos['Carga_de_calefacción'], fit=norm, color='red')
(mu, sigma) = stats.norm.fit(datos['Carga_de_calefacción'])
ax1.set(title='Distribución Normal ( $\mu = \$ {:.2f}$  and  $\sigma = \$ {:.2f}$ )'.format(mu, sigma))

# PDF
ax2 = plt.subplot(222)
stats.probplot(datos['Carga_de_calefacción'], plot=plt)

fig = plt.figure()
```

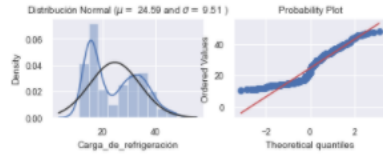


<Figure size 432x288 with 0 Axes>

```
In [13]: # Comprueba La distribución normal de La carga derefrigeración con un histograma y máxima probabilidad gaussiana
ax1 = plt.subplot(221)
sns.distplot(datos['Carga_de_refrigeración'], fit=norm)
(mu, sigma) = stats.norm.fit(datos['Carga_de_refrigeración'])
ax1.set(title='Distribución Normal ( $\mu =$ {:.2f} and  $\sigma =$ {:.2f})'.format(mu, sigma))

# PDF
ax2 = plt.subplot(222)
stats.probplot(datos['Carga_de_refrigeración'], plot=plt)

fig = plt.figure()
```

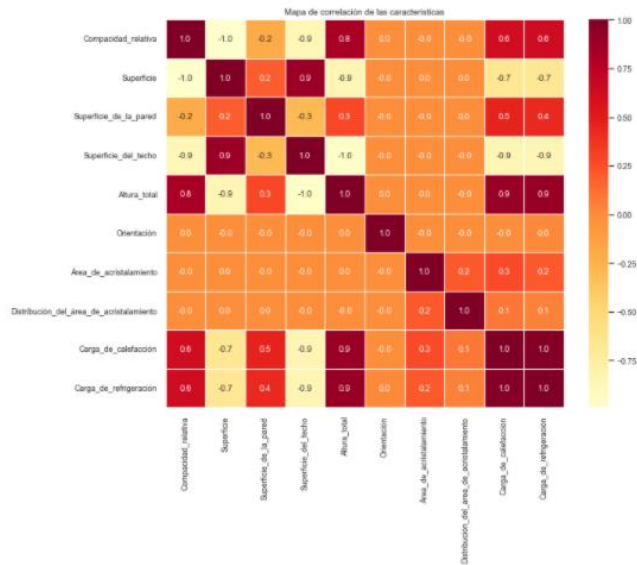


<Figure size 432x288 with 0 Axes>

Coefficiente de correlación de Pearson

```
In [14]: # mapa de correlación de Las características
f,ax = plt.subplots(figsize=(10, 8))
ax.set_title('Mapa de correlación de las características')
sns.heatmap(datos.corr(), annot=True, linewidths=.5, fmt= '.1f',ax=ax,cmap="YlOrRd")
```

Out[14]: <AxesSubplot:title={'center':'Mapa de correlación de las características'}>



```
In [15]: # Cambia el formato del número en Las correlaciones
pd.set_option('display.float_format',lambda x: '{:,.2f}'.format(x) if abs(x) < 10000 else '{:,.0f}'.format(x))
datos.corr()
```

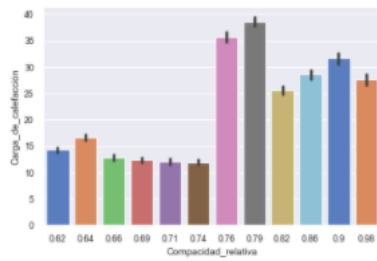
Out[15]:

	Compacidad_relativa	Superficie	Superficie_de_la_pared	Superficie_del_techo	Altura_total	Orientación	Área_de_acrista
Compacidad_relativa	1.00	-0.99	-0.20	-0.87	0.83	0.00	
Superficie	-0.99	1.00	0.20	0.88	-0.86	-0.00	
Superficie_de_la_pared	-0.20	0.20	1.00	-0.29	0.28	-0.00	
Superficie_del_techo	-0.87	0.88	-0.29	1.00	-0.97	-0.00	
Altura_total	0.83	-0.86	0.28	-0.97	1.00	0.00	
Orientación	0.00	-0.00	-0.00	-0.00	0.00	1.00	
Área_de_acristalamiento	-0.00	0.00	-0.00	-0.00	0.00	-0.00	1.00
Distribución_del_área_de_acristalamiento	-0.00	0.00	0.00	-0.00	-0.00	-0.00	-0.00
Carga_de calefacción	0.62	-0.66	0.46	-0.86	0.89	-0.00	
Carga_de_refrigeración	0.63	-0.67	0.43	-0.86	0.90	0.01	

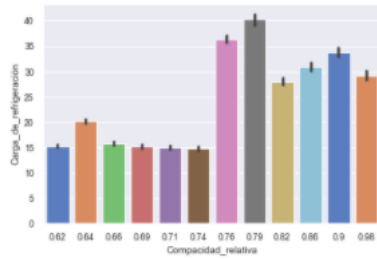
```
In [16]: print ("Los valores únicos son:\n",datos.nunique())
```

```
Los valores únicos son:
Compacidad_relativa      12
Superficie                12
Superficie_de_la_pared   7
Superficie_del_techo     4
Altura_total              2
Orientación               4
Área_de_acristalamiento  4
Distribución_del_área_de_acristalamiento  6
Carga_de calefacción     586
Carga_de_refrigeración   636
dtype: int64
```

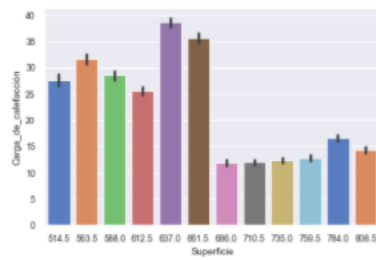
In [17]: # Grafico de barras para tener una idea sobre La distribución de La compacidad relativa frente a La carga de calefacción
 p = sns.barplot(data=datos, x = 'Compacidad_relativa', y = 'Carga_de_calefacción', palette='muted')



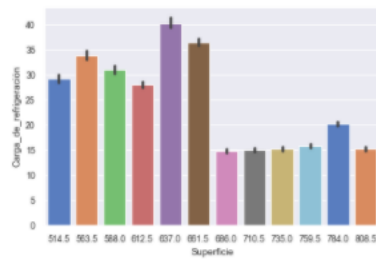
In [18]: # Grafico de barras para tener una idea sobre La distribución de La compacidad relativa frente a La carga de refrigeración
 p = sns.barplot(data=datos, x = 'Compacidad_relativa', y = 'Carga_de_refrigeración', palette='muted')



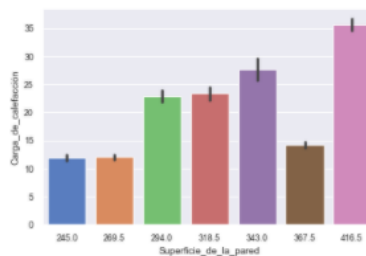
In [19]: # Grafico de barras para tener una idea sobre el área de superficie frente a La carga de calefacción
 p = sns.barplot(data=datos, x = 'Superficie', y = 'Carga_de_calefacción', palette='muted')



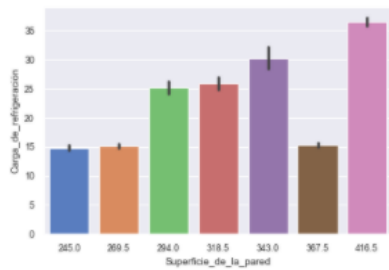
In [20]: # Grafico de barras para tener una idea sobre el área de superficie frente a La carga de refrigeración
 p = sns.barplot(data=datos, x = 'Superficie', y = 'Carga_de_refrigeración', palette='muted')



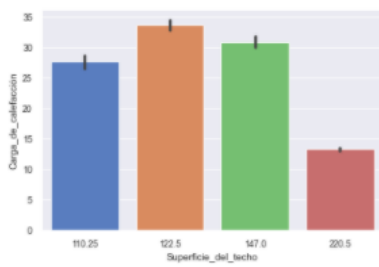
In [21]: # Grafico de barras para tener una idea sobre La superficie de La pared vs La carga de calefacción
 p = sns.barplot(data=datos, x = 'Superficie_de_la_pared', y = 'Carga_de_calefacción', palette='muted')



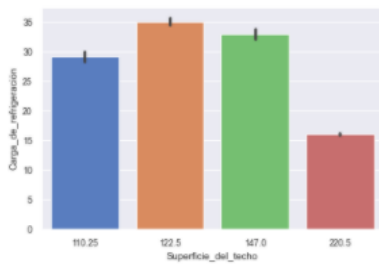
```
In [22]: # Grafico de barras para tener una idea sobre La superficie de la pared vs La carga de refrigeración
p = sns.barplot(data=datos, x = 'Superficie_de_la_pared', y = 'Carga_de_refrigeración', palette='muted')
```



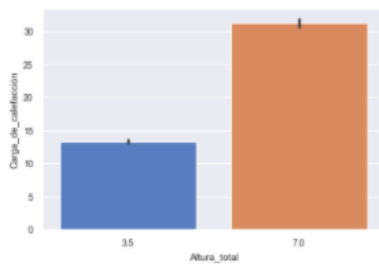
```
In [23]: # Grafico de barras para tener una idea sobre La superficie del techo vs La carga de calefacción
p = sns.barplot(data=datos, x = 'Superficie_del_techo', y = 'Carga_de_calefacción', palette='muted')
```



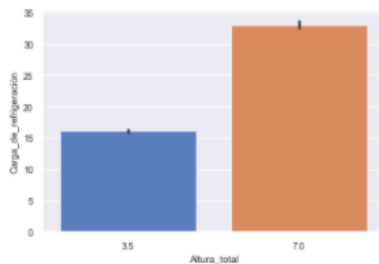
```
In [24]: # Grafico de barras para tener una idea sobre La superficie del techo vs La carga de refrigeración
p = sns.barplot(data=datos, x = 'Superficie_del_techo', y = 'Carga_de_refrigeración', palette='muted')
```



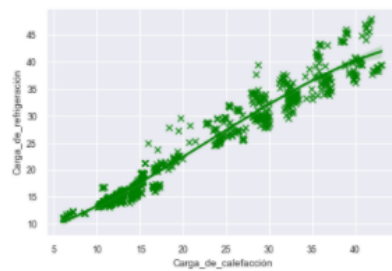
```
In [25]: # Grafico de barras para tener una idea sobre La altura total vs La carga de calefacción
p = sns.barplot(data=datos, x = 'Altura_total', y = 'Carga_de_calefacción', palette='muted')
```



```
In [26]: # Grafico de barras para tener una idea sobre La altura total vs La carga de refrigeración
p = sns.barplot(data=datos, x = 'Altura_total', y = 'Carga_de_refrigeración', palette='muted')
```




```
In [27]: sns.regplot(x = datos.Carga_de calefacción,y = datos.Carga_de refrigeración,order=3,marker='x',color='green')
plt.show()
```



División del conjunto de datos para la predicción ⌚

```
In [28]: #Normaliza Las entradas y divide La salida
from sklearn.preprocessing import Normalizer
nr = Normalizer(copy=False)

# Definimos (X) y (y)
x = datos.drop(['Carga_de calefacción', 'Carga_de refrigeración'], axis=1)
x = nr.fit_transform(x)
y = datos[['Carga_de calefacción', 'Carga_de refrigeración']]

print('Dimensión de las características:',x.shape)
print('Dimensión del objetivo:',y.shape)
```

```
In [29]: # División del conjunto de datos para prueba y emtenamiento
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.25, random_state = 10)

print('X Train:',x_train.shape)
print('X Test:',x_test.shape)
print('Y Train:',y_train.shape)
print('Y Test:',y_test.shape)

X Train: (576, 8)
X Test: (192, 8)
Y Train: (576, 2)
Y Test: (192, 2)
```



Contrucción de los modelos 📄

Regresión lineal

```
In [31]: # carga el modelo de regresión
from sklearn.linear_model import LinearRegression

# crea el modelo de regresión lineal
Regresión = LinearRegression()

# Ajuste de Los datos de (x) y (y) para La regresión
Regresión.fit(x_train, y_train)

# Predice La respuesta del modelo
y_pred1 = Regresión.predict(x_test)

acc_linreg = round(Regresión.score(x_test,y_test)* 100, 2)
print("Precisión del modelo de regresión lineal:",acc_linreg,"%")

Precisión del modelo de regresión lineal: 92.07 %
```

```
In [32]: predicción_calefacción1 = pd.DataFrame({'Valor Actual': y_test['Carga_de_calefacción'], 'Predicción': y_pred1[:,0]})
predicción_calefacción1
```

Out[32]:

	Valor Actual	Predicción
568	19.52	17.47
620	16.76	18.40
456	12.32	11.65
197	28.05	27.42
714	19.00	17.94
...
613	15.29	16.87
562	14.42	15.63
587	31.84	32.95
413	12.28	12.45
487	32.46	31.13

```
In [33]: predicción_refrigeración1 = pd.DataFrame({'Valor Actual': y_test["Carga_de_refrigeración"], 'Predicción': y_pred1[:,1]})
predicción_refrigeración1
```

Out[33]:

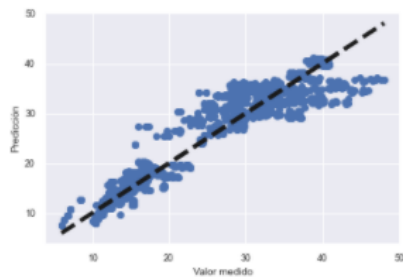
	Valor Actual	Predicción
568	22.72	19.25
620	17.36	19.94
456	14.92	14.33
197	33.91	29.69
714	22.25	19.61
...
613	17.89	18.80
562	16.87	17.94
587	38.88	34.78
413	14.74	15.30
487	35.62	32.64

192 rows x 2 columns

```
In [34]: from sklearn.model_selection import cross_val_predict

# Genera estimaciones de la validación cruzada para cada punto de datos de entrada
predicted = cross_val_predict(Regresión, x, y, cv=10)

# Traza las salidas previstas con respecto a las medidas
fig, ax = plt.subplots()
ax.scatter(y, predicted)
ax.plot([y.min(), y.max()], [y.min(), y.max()], 'k--', lw=4)
ax.set_xlabel('Valor medido')
ax.set_ylabel('Predicción')
plt.show()
```



```
In [35]: # Métricas
print('Error absoluto medio(MAE):',mean_absolute_error(y_test, y_pred1, sample_weight=None, multioutput='uniform_average'))
print('Desviación:',explained_variance_score(y_test, y_pred1, multioutput='variance_weighted'))
print('Error cuadrático medio (MSE):',mean_squared_error(y_test, y_pred1, sample_weight=None, multioutput='uniform_average'))
print('Raíz del Error cuadrático medio (RMSE):',np.sqrt(mean_squared_error(y_test, y_pred1, sample_weight=None, multioutput='uniform_average'))
print('R cuadrado:',r2_score(y_test, y_pred1, sample_weight=None, multioutput='uniform_average'))
```

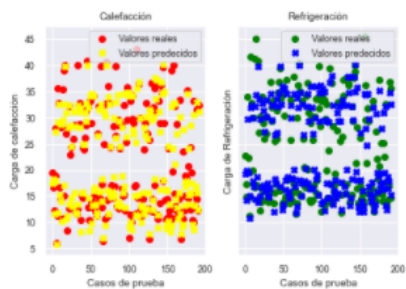
```
Error absoluto medio(MAE): 1.879494628906249
Desviación: 0.9217726182276913
Error cuadrático medio (MSE): 7.242145130505972
Raíz del Error cuadrático medio (RMSE): 2.691123395629783
R cuadrado: 0.9206870050895783
```

```
In [36]: f, (ax1, ax2) = plt.subplots(1, 2, sharey=True)

# Visualiza La salida de carga de calefacción
plt.figure(figsize = (10,10))
ax1.plot(range(0,len(x_test)),y_test.iloc[:,0], 'o',color='red',label = 'Valores reales')
ax1.plot(range(0,len(x_test)),y_pred1[:,0], 'x',color='yellow',label = 'Valores predcidos')
ax1.set_xlabel('Casos de prueba')
ax1.set_ylabel('Carga de calefacción')
ax1.set_title('Calefacción')
ax1.legend(loc = 'upper right')

# Visualiza La salida de carga de refrigeración
plt.figure(figsize = (10,10))
ax2.plot(range(0,len(x_test)),y_test.iloc[:,1], 'o',color='green',label = 'Valores reales')
ax2.plot(range(0,len(x_test)),y_pred1[:,1], 'x',color='blue',label = 'Valores predcidos')
ax2.set_xlabel('Casos de prueba')
ax2.set_ylabel('Carga de Refrigeración')
ax2.set_title('Refrigeración')
ax2.legend(loc = 'upper right')

plt.show()
```



<Figure size 720x720 with 0 Axes>

<Figure size 720x720 with 0 Axes>

Decision Tree Regressor

```
In [37]: #Carga el modelo de decision tree regressor
from sklearn.tree import DecisionTreeRegressor

# crea el modelo de decision tree regressor
dt_model = DecisionTreeRegressor(random_state=123)

# Ajuste de los datos de (x) y (y) para la regresión
dt_model.fit(x_train, y_train)

# Predice la respuesta del modelo
y_pred2 = dt_model.predict(x_test)

acc_dtr = round(dt_model.score(x_test,y_test)* 100, 2)
print(" Precisión del modelo de regresión de decision tree regressor :",acc_dtr,"%")
```

Precisión del modelo de regresión de decision tree regressor : 96.65 %

```
In [38]: predicción_calefacción2 = pd.DataFrame({'Valor Actual': y_test['Carga_de_calefacción'], 'Predicción': y_pred2[:,0]})
predicción_calefacción2
```

Out[38]:

	Valor Actual	Predicción
588	19.82	19.38
620	16.76	15.09
456	12.32	12.03
197	28.05	28.86
714	19.00	19.13
...
613	15.29	15.18
562	14.42	14.33
587	31.84	32.13
413	12.28	12.29
487	32.46	33.28

192 rows x 2 columns

```
In [39]: predicción_refrigeración2 = pd.DataFrame({'Valor Actual': y_test["Carga_de_refrigeración"], 'Predicción': y_pred2[:,1]})
predicción_refrigeración2
```

Out[39]:

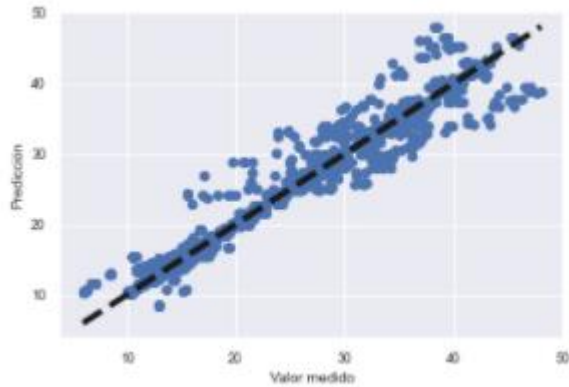
	Valor Actual	Predicción
568	22.72	22.73
620	17.36	17.20
456	14.92	13.79
197	33.91	32.83
714	22.25	21.54
...
613	17.89	17.83
562	18.87	17.23
587	36.86	32.28
413	14.74	15.40
487	35.82	33.16

192 rows x 2 columns

```
In [40]: from sklearn.model_selection import cross_val_predict

# Genera estimaciones de la validación cruzada para cada punto de datos de entrada
predicted = cross_val_predict(dt_model, x, y, cv=10)

# Traza las salidas previstas con respecto a las medidas
fig, ax = plt.subplots()
ax.scatter(y, predicted)
ax.plot([y.min(), y.max()], [y.min(), y.max()], 'k--', lw=4)
ax.set_xlabel('Valor medido')
ax.set_ylabel('Predicción')
plt.show()
```



```
In [41]: # Métricas sin optimización

print('Error absoluto medio(MAE):',mean_absolute_error(y_test, y_pred2, sample_weight=None, multioutput='uniform_average'))
print('Desviación:',explained_variance_score(y_test, y_pred2, multioutput='variance_weighted'))
print('Error cuadrático medio (MSE):',mean_squared_error(y_test, y_pred2, sample_weight=None, multioutput='uniform_average'))
print('Raíz del Error cuadrático medio (RMSE):',np.sqrt(mean_squared_error(y_test, y_pred2, sample_weight=None, multioutput='uniform_average')))
print('R cuadrado:',r2_score(y_test, y_pred2, sample_weight=None, multioutput='uniform_average'))
```

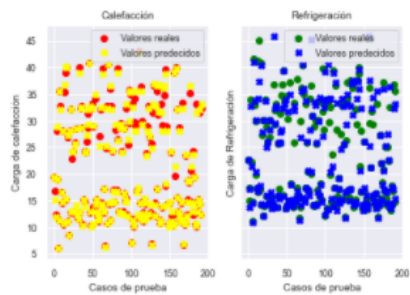
Error absoluto medio(MAE): 0.8427604166666666
Desviación: 0.9683323767600238
Error cuadrático medio (MSE): 2.9239031250000003
Raíz del Error cuadrático medio (RMSE): 1.7099424332415405
R cuadrado: 0.966513701581212

```
In [42]: f, (ax1, ax2) = plt.subplots(1, 2, sharey=True)

# Visualiza La salida de carga de calefacción sin optimizar
plt.figure(figsize = (10,10))
ax1.plot(range(0,len(x_test)),y_test.iloc[:,0],'o',color='red',label = 'Valores reales')
ax1.plot(range(0,len(x_test)),y_pred2[:,0],'x',color='yellow',label = 'Valores predcidos')
ax1.set_xlabel('Casos de prueba')
ax1.set_ylabel('Carga de calefacción')
ax1.set_title('Calefacción')
ax1.legend(loc = 'upper right')

# Visualiza La salida de carga de refrigeración sin optimizar
plt.figure(figsize = (10,10))
ax2.plot(range(0,len(x_test)),y_test.iloc[:,1],'o',color='green',label = 'Valores reales')
ax2.plot(range(0,len(x_test)),y_pred2[:,1],'x',color='blue',label = 'Valores predcidos')
ax2.set_xlabel('Casos de prueba')
ax2.set_ylabel('Carga de Refrigeración')
ax2.set_title('Refrigeración')
ax2.legend(loc = 'upper right')

plt.show()
```



<Figure size 720x720 with 0 Axes>

<Figure size 720x720 with 0 Axes>

```
In [43]: # Encuentra Los mejores parámetros de optimización para el modelo de decision tree regressor
```

```
f, (ax1, ax2) = plt.subplots(1, 2, sharey=True)

# Max Profundidad
dt_acc = []
dt_depth = range(1,11)
for i in dt_depth:
    dt = DecisionTreeRegressor(random_state=123, max_depth=i)
    dt.fit(x_train, y_train)
    dt_acc.append(dt.score(x_test, y_test))
ax1.plot(dt_depth,dt_acc)
ax1.set_title('Max Profundidad')

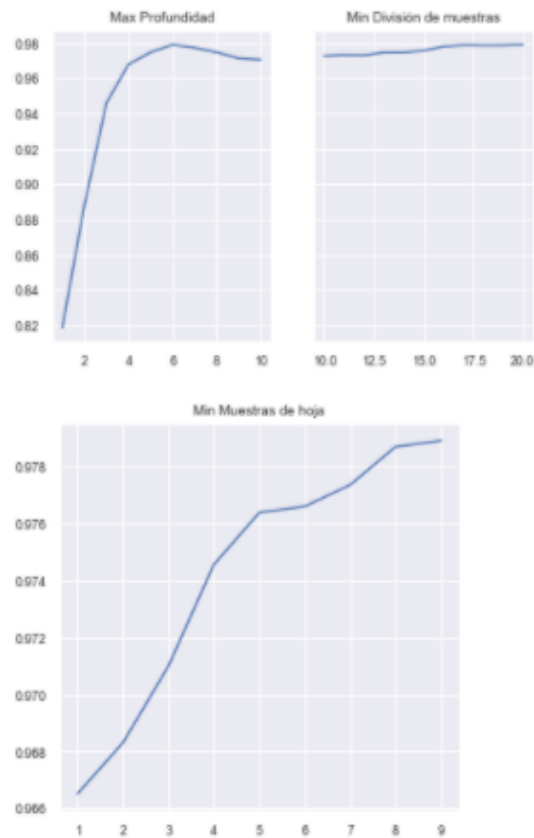
# Min División de muestras
dt_acc = []
dt_samples_split = range(10,21)
for i in dt_samples_split:
    dt = DecisionTreeRegressor(random_state=123, min_samples_split=i)
    dt.fit(x_train, y_train)
    dt_acc.append(dt.score(x_test, y_test))
ax2.plot(dt_samples_split,dt_acc)
ax2.set_title('Min División de muestras')

plt.show()

# Min Muestras de hoja
plt.figure(figsize = (5,5))
dt_acc = []
dt_samples_leaf = range(1,10)
for i in dt_samples_leaf:
    dt = DecisionTreeRegressor(random_state=123, min_samples_leaf=i)
    dt.fit(x_train, y_train)
    dt_acc.append(dt.score(x_test, y_test))

plt.plot(dt_samples_leaf,dt_acc)
plt.title('Min Muestras de hoja')

plt.show()
```



In [44]: # Parámetros de optimización de Decision Tree Regressor

```
from sklearn.model_selection import GridSearchCV
parameters = {'max_depth': [7,8,9],
              'min_samples_split': [16,17,18],
              'min_samples_leaf': [6,7,8]}
```

```
# Crea un nuevo modelo utilizando GridSearch
dt_random = GridSearchCV(dt_model, parameters, cv=10)
```

```
# Aplica el modelo
dt_random.fit(x_train, y_train)
```

Out[44]: GridSearchCV(cv=10, estimator=DecisionTreeRegressor(random_state=123),
param_grid={'max_depth': [7, 8, 9], 'min_samples_leaf': [6, 7, 8],
'min_samples_split': [16, 17, 18]})

In [45]: # Impresión de Los mejores parámetros
dt_random.best_params_

Out[45]: {'max_depth': 8, 'min_samples_leaf': 6, 'min_samples_split': 18}

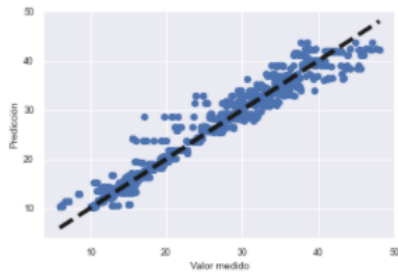
In [46]: # Valores predichos

```
y_pred2new = dt_random.best_estimator_.predict(x_test)
```

In [47]: from sklearn.model_selection import cross_val_predict

```
# Genera estimaciones de la validación cruzada para cada punto de datos de entrada
predicted = cross_val_predict(dt_random, x, y, cv=10)
```

```
# Traza Las salidas previstas con respecto a Las medidas
fig, ax = plt.subplots()
ax.scatter(y, predicted)
ax.plot([y.min(), y.max()], [y.min(), y.max()], 'k--', lw=4)
ax.set_xlabel('Valor medido')
ax.set_ylabel('Predicción')
plt.show()
```



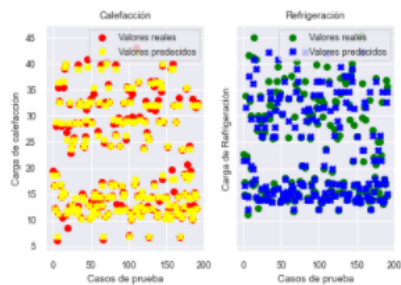
```
In [48]: # Métricas optimizadas
print('Error absoluto medio(MAE):',mean_absolute_error(y_test, y_pred2new, sample_weight=None, multioutput='uniform_average'))
print('Desviación:',explained_variance_score(y_test, y_pred2new, multioutput='variance_weighted'))
print('Error cuadrático medio (MSE):',mean_squared_error(y_test, y_pred2new, sample_weight=None, multioutput='uniform_average'))
print('Raíz del Error cuadrático medio (RMSE):',np.sqrt(mean_squared_error(y_test, y_pred2new, sample_weight=None, multioutput='uniform_average'))
print('R cuadrado:',r2_score(y_test, y_pred2new, sample_weight=None, multioutput='uniform_average'))
```

Error absoluto medio(MAE): 0.8215576987350885
 Desviación: 0.9797979999820037
 Error cuadrático medio (MSE): 1.0618527332313217
 Raíz del Error cuadrático medio (RMSE): 1.3644972455931605
 R cuadrado: 0.9789169597062091

```
In [49]: f, (ax1, ax2) = plt.subplots(1, 2, sharey=True)
# Visualización de la carga de calefacción después de la optimización
plt.figure(figsize = (5,5))
ax1.plot(range(0,len(x_test)),y_test.iloc[:,0], 'o',color='red',label = 'Valores reales')
ax1.plot(range(0,len(x_test)),y_pred2new[:,0], 'x',color='yellow',label = 'Valores predcidos')
ax1.set_xlabel('Casos de prueba')
ax1.set_ylabel('Carga de calefacción')
ax1.set_title('Calefacción')
ax1.legend(loc = 'upper right')

# Visualiza La salida de carga de refrigeración después de la optimización
plt.figure(figsize = (10,10))
ax2.plot(range(0,len(x_test)),y_test.iloc[:,1], 'o',color='green',label = 'Valores reales')
ax2.plot(range(0,len(x_test)),y_pred2new[:,1], 'x',color='blue',label = 'Valores predcidos')
ax2.set_xlabel('Casos de prueba')
ax2.set_ylabel('Carga de Refrigeración')
ax2.set_title('Refrigeración')
ax2.legend(loc = 'upper right')

plt.show()
```



<Figure size 360x360 with 0 Axes>

<Figure size 720x720 with 0 Axes>

Random Forest Regressor

```
In [50]: # Carga el modelo de random forest regressor
from sklearn.ensemble import RandomForestRegressor

# crea el modelo de random forest regressor
reg= RandomForestRegressor(n_estimators = 200, random_state = 42)

# Ajuste de los datos de (x) y (y) para la regresión
reg.fit(x_train, y_train)
y_pred3 = reg.predict(x_test)

# Predice la respuesta del modelo
acc_rfr = round(reg.score(x_test,y_test)* 100, 2)
print(" Precisión del modelo de regresión de Random Forest :",acc_rfr,"%")
```

Precisión del modelo de regresión de Random Forest : 97.83 %

```
In [51]: predicción_calefacción3 = pd.DataFrame({'Valor Actual': y_test['Carga_de_calefacción'], 'Predicción': y_pred3[:,0]})
predicción_calefacción3
```

Out[51]:

	Valor Actual	Predicción
568	19.52	18.89
620	16.76	15.74
456	12.32	12.08
197	28.05	29.14
714	19.00	18.86
...
613	15.29	15.22
562	14.42	14.44
587	31.84	32.20
413	12.28	12.35
487	32.46	32.65

192 rows x 2 columns

```
In [52]: predicción_refrigeración3 = pd.DataFrame({'Valor Actual': y_test["Carga_de_refrigeración"], 'Predicción': y_pred3[:,1]})
predicción_refrigeración3
```

Out[52]:

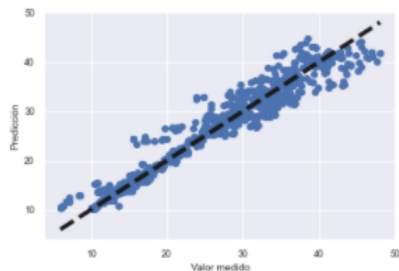
	Valor Actual	Predicción
568	22.72	22.04
620	17.36	17.11
456	14.92	14.48
197	33.91	31.47
714	22.25	21.55
...
613	17.89	17.81
562	16.87	17.17
587	36.86	31.99
413	14.74	14.98
487	35.62	33.81

192 rows x 2 columns

```
In [53]: from sklearn.model_selection import cross_val_predict

# Genera estimaciones de la validación cruzada para cada punto de datos de entrada
predicted = cross_val_predict(reg, X, y, cv=10)

# Traza las salidas previstas con respecto a las medidas
fig, ax = plt.subplots()
ax.scatter(y, predicted)
ax.plot([y.min(), y.max()], [y.min(), y.max()], 'k--', lw=4)
ax.set_xlabel('Valor medido')
ax.set_ylabel('Predicción')
plt.show()
```



```
In [54]: # Métricas
```

```
print('Error absoluto medio(MAE):',mean_absolute_error(y_test, y_pred3, sample_weight=None, multioutput='uniform_average'))
print('Desviación:',explained_variance_score(y_test, y_pred3, multioutput='variance_weighted'))
print('Error cuadrático medio (MSE):',mean_squared_error(y_test, y_pred3, sample_weight=None, multioutput='uniform_average'))
print('Raíz del Error cuadrático medio (RMSE):',np.sqrt(mean_squared_error(y_test, y_pred3, sample_weight=None, multioutput='uniform_average'))
print('R cuadrado:',r2_score(y_test, y_pred3, sample_weight=None, multioutput='uniform_average'))
```

```
Error absoluto medio(MAE): 0.7490816406250009
Desviación: 0.9793553755100953
Error cuadrático medio (MSE): 1.9023124965169262
Raíz del Error cuadrático medio (RMSE): 1.379243450779059
R cuadrado: 0.9782649267370356
```

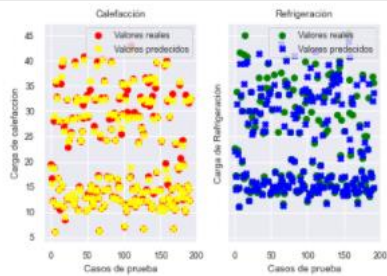


```
In [55]: f, (ax1, ax2) = plt.subplots(1, 2, sharey=True)

# Visualiza La salida de carga de calefacción
plt.figure(figsize = (10,10))
ax1.plot(range(0,len(x_test)),y_test.iloc[:,0], 'o',color='red',label = 'Valores reales')
ax1.plot(range(0,len(x_test)),y_pred3[:,0], 'x',color='yellow',label = 'Valores predcidos')
ax1.set_xlabel('casos de prueba')
ax1.set_ylabel('Carga de calefacción')
ax1.set_title('Calefacción')
ax1.legend(loc = 'upper right')

# Visualiza La salida de carga de refrigeración
plt.figure(figsize = (10,10))
ax2.plot(range(0,len(x_test)),y_test.iloc[:,1], 'o',color='green',label = 'Valores reales')
ax2.plot(range(0,len(x_test)),y_pred3[:,1], 'x',color='blue',label = 'Valores predcidos')
ax2.set_xlabel('casos de prueba')
ax2.set_ylabel('Carga de Refrigeración')
ax2.set_title('Refrigeración')
ax2.legend(loc = 'upper right')

plt.show()
```



<Figure size 720x720 with 0 Axes>

<Figure size 720x720 with 0 Axes>

```
In [56]: # Encuentra Los mejores parámetros de optimización para el modelo de random forest regressor
```

```
f, axarr = plt.subplots(2, 2)

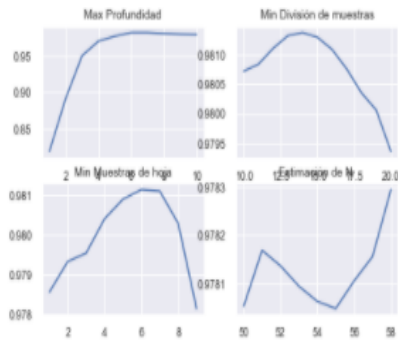
# Max Profundidad
rf_acc = []
rf_depth = range(1,11)
for i in rf_depth:
    rf = RandomForestRegressor(random_state=123, max_depth=i)
    rf.fit(x_train, y_train)
    rf_acc.append(rf.score(x_test, y_test))
axarr[0, 0].plot(rf_depth,rf_acc)
axarr[0, 0].set_title('Max Profundidad')

#Min División de muestras
rf_acc = []
rf_samples_split = range(10,21)
for i in rf_samples_split:
    rf = RandomForestRegressor(random_state=123, min_samples_split=i)
    rf.fit(x_train, y_train)
    rf_acc.append(rf.score(x_test, y_test))
axarr[0, 1].plot(rf_samples_split,rf_acc)
axarr[0, 1].set_title('Min División de muestras')

#Min Muestras de hojas
rf_acc = []
rf_samples_leaf = range(1,10)
for i in rf_samples_leaf:
    rf = RandomForestRegressor(random_state=123, min_samples_leaf=i)
    rf.fit(x_train, y_train)
    rf_acc.append(rf.score(x_test, y_test))
axarr[1, 0].plot(rf_samples_leaf,rf_acc)
axarr[1, 0].set_title('Min Muestras de hoja')

#Estimación de N
rf_acc = []
rf_estimators = range(50,59)
for i in rf_estimators:
    rf = RandomForestRegressor(random_state=123, n_estimators=i)
    rf.fit(x_train, y_train)
    rf_acc.append(rf.score(x_test, y_test))
axarr[1, 1].plot(rf_estimators,rf_acc)
axarr[1, 1].set_title('Estimación de N ')

plt.show()
```



```
In [57]: # Parametros de Optimización de Random forest Regresor
from sklearn.model_selection import GridSearchCV
parameters = {'max_depth' : [6,7,8],
              'min_samples_split': [11,12,13],
              'min_samples_leaf' : [4,5,6],
              'n_estimators' : [49,50,51]}

# Crea un nuevo modelo usando GridSearch
rf_random = GridSearchCV(reg, parameters, cv=10)

#aplica el modelo
rf_random.fit(x_train, y_train)
```

```
Out[57]: GridSearchCV(cv=10,
                    estimator=RandomForestRegressor(n_estimators=200, random_state=42),
                    param_grid={'max_depth': [6, 7, 8], 'min_samples_leaf': [4, 5, 6],
                                'min_samples_split': [11, 12, 13],
                                'n_estimators': [49, 50, 51]})
```

```
In [58]: # Impresión de Los mejores parámetros
rf_random.best_params_
```

```
Out[58]: {'max_depth': 7,
          'min_samples_leaf': 4,
          'min_samples_split': 12,
          'n_estimators': 50}
```

```
In [59]: # Valores predcidos
y_pred3new = rf_random.best_estimator_.predict(x_test)
```

```
In [60]: # Métricas optimizadas

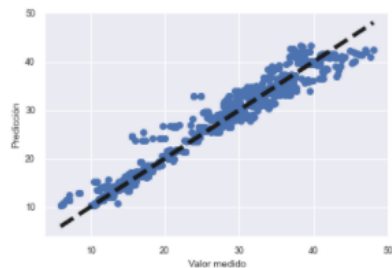
print('Error absoluto medio(MAE):',mean_absolute_error(y_test, y_pred3new, sample_weight=None, multioutput='uniform_average'))
print('Desviación:',explained_variance_score(y_test, y_pred3new, multioutput='variance_weighted'))
print('Error cuadratico medio (MSE):',mean_squared_error(y_test, y_pred3new, sample_weight=None, multioutput='uniform_average'))
print('Raiz del Error cuadratico medio (RMSE):',np.sqrt(mean_squared_error(y_test, y_pred3new, sample_weight=None, multioutput='uniform_average'))
print('R cuadrado:',r2_score(y_test, y_pred3new, sample_weight=None, multioutput='uniform_average'))
```

Error absoluto medio(MAE): 0.791222841727711
Desviación: 0.9813459327807216
Error cuadratico medio (MSE): 1.7185460223977014
Raiz del Error cuadratico medio (RMSE): 1.3109332638993112
R cuadrado: 0.9805231835361063

```
In [76]: from sklearn.model_selection import cross_val_predict

# Genera estimaciones de La validación cruzada para cada punto de datos de entrada
predicted = cross_val_predict(rf_random, x, y, cv=10)

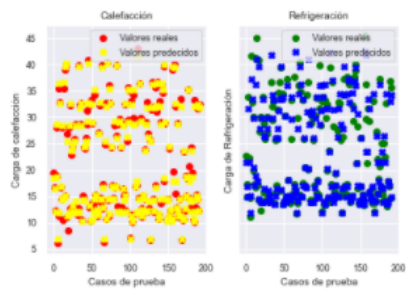
# Traza Las salidas previstas con respecto a Las medidas
fig, ax = plt.subplots()
ax.scatter(y, predicted)
ax.plot([y.min(), y.max()], [y.min(), y.max()], 'k--', lw=4)
ax.set_xlabel('Valor medido')
ax.set_ylabel('Predicción')
plt.show()
```



```
In [61]: f, (ax1, ax2) = plt.subplots(1, 2, sharey=True)
# Visualización de La carga de calefacción después de la optimización
plt.figure(figsize = (5,5))
ax1.plot(range(0,len(x_test)),y_test.iloc[:,0], 'o',color='red',label = 'Valores reales')
ax1.plot(range(0,len(x_test)),y_pred3new[:,0], 'x',color='yellow',label = 'Valores predcidos')
ax1.set_xlabel('Casos de prueba')
ax1.set_ylabel('Carga de calefacción')
ax1.set_title('Calefacción')
ax1.legend(loc = 'upper right')

# Visualiza La salida de carga de refrigeración después de la optimización
plt.figure(figsize = (10,10))
ax2.plot(range(0,len(x_test)),y_test.iloc[:,1], 'o',color='green',label = 'Valores reales')
ax2.plot(range(0,len(x_test)),y_pred3new[:,1], 'x',color='blue',label = 'Valores predcidos')
ax2.set_xlabel('Casos de prueba')
ax2.set_ylabel('Carga de Refrigeración')
ax2.set_title('Refrigeración')
ax2.legend(loc = 'upper right')

plt.show()
```



Extra Trees Regressor

```
In [62]: # Carga el clasificador de Extra Trees Regressor
from sklearn.ensemble import ExtraTreesRegressor

# crea el modelo de regresión
etr_model = ExtraTreesRegressor(random_state=123)

# Ajuste de Los datos de (x) y (y) para La regresión
etr_model.fit(x_train, y_train)
y_pred4 = etr_model.predict(x_test)

# Predice La respuesta del modelo
acc_etr = round(etr_model.score(x_test,y_test)* 100, 2)
print(" Precisión del modelo de regresion de Extra Trees :",acc_etr,"%")

Precisión del modelo de regresion de Extra Trees : 97.6 %
```

```
In [63]: predicción_calefacción4 = pd.DataFrame({'Valor Actual': y_test['Carga_de_calefacción'], 'Predicción': y_pred4[:,0]})
predicción_calefacción4
```

Out[63]:

	Valor Actual	Predicción
568	19.52	19.36
620	16.76	15.57
456	12.32	12.16
197	28.05	29.17
714	19.00	18.78
...
613	15.29	15.21
562	14.42	14.51
587	31.84	32.02
413	12.28	12.40
487	32.46	33.26

192 rows x 2 columns

```
In [64]: predicción_refrigeración4 = pd.DataFrame({'Valor Actual': y_test["Carga_de_refrigeración"], 'Predicción': y_pred4[:,1]})
predicción_refrigeración4
```

Out[64]:

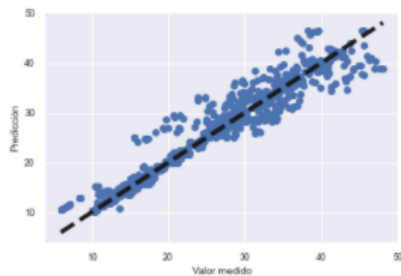
	ValorActual	Predicción
568	22.72	22.71
620	17.36	17.10
456	14.92	14.64
197	33.91	31.33
714	22.25	21.56
...
613	17.89	17.81
562	16.87	17.13
587	38.88	31.92
413	14.74	15.00
487	35.82	33.22

192 rows x 2 columns

```
In [65]: from sklearn.model_selection import cross_val_predict

# Genera estimaciones de la validación cruzada para cada punto de datos de entrada
predicted = cross_val_predict(etr_model, X, y, cv=10)

# Traza las salidas previstas con respecto a las medidas
fig, ax = plt.subplots()
ax.scatter(y, predicted)
ax.plot([y.min(), y.max()], [y.min(), y.max()], 'k--', lw=4)
ax.set_xlabel('Valor medido')
ax.set_ylabel('Predicción')
plt.show()
```



```
In [66]: # Métricas

print('Error absoluto medio(MAE):',mean_absolute_error(y_test, y_pred4, sample_weight=None, multioutput='uniform_average'))
print('Desviación:',explained_variance_score(y_test, y_pred4, multioutput='variance_weighted'))
print('Error cuadrático medio (MSE):',mean_squared_error(y_test, y_pred4, sample_weight=None, multioutput='uniform_average'))
print('Raíz del Error cuadrático medio (RMSE):',np.sqrt(mean_squared_error(y_test, y_pred4, sample_weight=None, multioutput='uniform_average')))
print('R cuadrado:',r2_score(y_test, y_pred4, sample_weight=None, multioutput='uniform_average'))
```

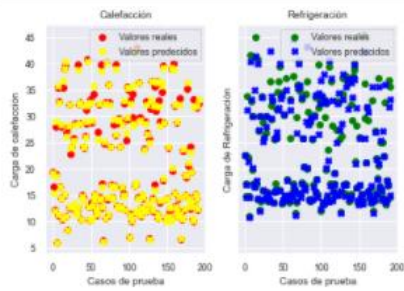
Error absoluto medio(MAE): 0.7357002604166669
Desviación: 0.9772375326944949
Error cuadrático medio (MSE): 2.096533015338534
Raíz del Error cuadrático medio (RMSE): 1.447940957131379
R cuadrado: 0.9760068668779631

```
In [67]: f, (ax1, ax2) = plt.subplots(1, 2, sharey=True)

# Visualiza la salida de carga de calefacción
plt.figure(figsize = (10,10))
ax1.plot(range(0,len(x_test)),y_test.iloc[:,0],'o',color='red',label = 'Valores reales')
ax1.plot(range(0,len(x_test)),y_pred4[:,0],'x',color='yellow',label = 'Valores predcidos')
ax1.set_xlabel('Casos de prueba')
ax1.set_ylabel('Carga de calefacción')
ax1.set_title('Calefacción')
ax1.legend(loc = 'upper right')

# Visualiza la salida de carga de refrigeración
plt.figure(figsize = (10,10))
ax2.plot(range(0,len(x_test)),y_test.iloc[:,1],'o',color='green',label = 'Valores reales')
ax2.plot(range(0,len(x_test)),y_pred4[:,1],'x',color='blue',label = 'Valores predcidos')
ax2.set_xlabel('Casos de prueba')
ax2.set_ylabel('Carga de Refrigeración')
ax2.set_title('Refrigeración')
ax2.legend(loc = 'upper right')

plt.show()
```



In [68]: # Encuentra los mejores parámetros de optimización para el modelo de extra trees regressor

```
f, axarr = plt.subplots(2, 2)

# Max Profundidad
etr_acc = []
etr_depth = range(1,11)
for i in etr_depth:
    etr = ExtraTreesRegressor(random_state=123, max_depth=i)
    etr.fit(x_train, y_train)
    etr_acc.append(etr.score(x_test, y_test))
axarr[0, 0].plot(etr_depth,etr_acc)
axarr[0, 0].set_title('Max Profundidad')

#Min División de muestras
etr_acc = []
etr_samples_split = range(16,26)
for i in etr_samples_split:
    etr = ExtraTreesRegressor(random_state=123, min_samples_split=i)
    etr.fit(x_train, y_train)
    etr_acc.append(etr.score(x_test, y_test))
axarr[0, 1].plot(etr_samples_split,etr_acc)
axarr[0, 1].set_title('Min División de muestras')

#Min Muestras de hojas
etr_acc = []
etr_samples_leaf = range(3,8)
for i in etr_samples_leaf:
    etr = ExtraTreesRegressor(random_state=123, min_samples_leaf=i)
    etr.fit(x_train, y_train)
    etr_acc.append(etr.score(x_test, y_test))

axarr[1, 0].plot(etr_samples_leaf,etr_acc)
axarr[1, 0].set_title('Min Muestras de hojas')

#Estimación de N
etr_acc = []
etr_estimators = range(40,46)
for i in etr_estimators:
    etr = ExtraTreesRegressor(random_state=123, n_estimators=i)
    etr.fit(x_train, y_train)
    etr_acc.append(etr.score(x_test, y_test))

axarr[1, 1].plot(etr_estimators,etr_acc)
axarr[1, 1].set_title('Estimación de N')

plt.show()
```



```
In [69]: # Parametros de Optimización de Extra trees regressor
from sklearn.model_selection import GridSearchCV
parameters = {'max_depth': [6,7,8],
             'min_samples_split': [19,20,21],
             'min_samples_leaf': [4,5,6],
             'n_estimators': [43,44,45]}

# Crea un nuevo modelo usando GridSearch
etr_random = GridSearchCV(etr_model, parameters, cv=10)

# aplica el modelo
etr_random.fit(X_train, y_train)
```

```
Out[69]: GridSearchCV(cv=10, estimator=ExtraTreesRegressor(random_state=123),
                    param_grid={'max_depth': [6, 7, 8], 'min_samples_leaf': [4, 5, 6],
                                 'min_samples_split': [19, 20, 21],
                                 'n_estimators': [43, 44, 45]})
```

```
In [70]: # Impresión de Los mejores parámetros
etr_random.best_params_
```

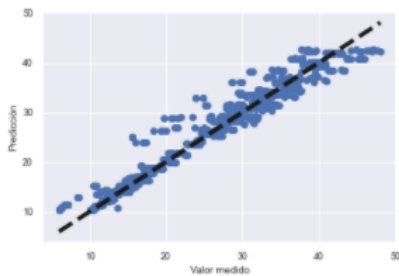
```
Out[70]: {'max_depth': 8,
          'min_samples_leaf': 4,
          'min_samples_split': 20,
          'n_estimators': 44}
```

```
In [71]: # Valores predcidos
y_pred4new = etr_random.best_estimator_.predict(x_test)
```

```
In [77]: from sklearn.model_selection import cross_val_predict

# Genera estimaciones de la validación cruzada para cada punto de datos de entrada
predicted = cross_val_predict(etr_random, x, y, cv=10)

# Traza Las salidas previstas con respecto a Las medidas
fig, ax = plt.subplots()
ax.scatter(y, predicted)
ax.plot([y.min(), y.max()], [y.min(), y.max()], 'k--', lw=4)
ax.set_xlabel('Valor medido')
ax.set_ylabel('Predicción')
plt.show()
```



```
In [72]: # Métricas optimizadas

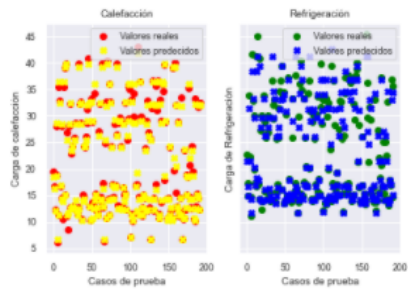
print('Error absoluto medio(MAE):',mean_absolute_error(y_test, y_pred4new, sample_weight=None, multioutput='uniform_average'))
print('Desviación:',explained_variance_score(y_test, y_pred4new, multioutput='variance_weighted'))
print('Error cuadrático medio (MSE):',mean_squared_error(y_test, y_pred4new, sample_weight=None, multioutput='uniform_average'))
print('Raíz del Error cuadrático medio (RMSE):',np.sqrt(mean_squared_error(y_test, y_pred4new, sample_weight=None, multioutput='uniform_average'))
print('R cuadrado:',r2_score(y_test, y_pred4new, sample_weight=None, multioutput='uniform_average'))
```

```
Error absoluto medio(MAE): 0.7628673413104428
Desviación: 0.9830792326382648
Error cuadrático medio (MSE): 1.5611992900631708
Raíz del Error cuadrático medio (RMSE): 1.2494796077020107
R cuadrado: 0.9823943335135821
```

```
In [74]: f, (ax1, ax2) = plt.subplots(1, 2, sharey=True)
# Visualización de La carga de calefacción después de La optimización
plt.figure(figsize = (5,5))
ax1.plot(range(0,len(x_test)),y_test.iloc[:,0], 'o', color='red', label = 'Valores reales')
ax1.plot(range(0,len(x_test)),y_pred4new[:,0], 'x', color='yellow', label = 'Valores predcidos')
ax1.set_xlabel('Casos de prueba')
ax1.set_ylabel('Carga de calefacción')
ax1.set_title('Calefacción')
ax1.legend(loc = 'upper right')

# Visualiza La salida de carga de refrigeración después de La optimización
plt.figure(figsize = (10,10))
ax2.plot(range(0,len(x_test)),y_test.iloc[:,1], 'o', color='green', label = 'Valores reales')
ax2.plot(range(0,len(x_test)),y_pred4new[:,1], 'x', color='blue', label = 'Valores predcidos')
ax2.set_xlabel('Casos de prueba')
ax2.set_ylabel('Carga de Refrigeración')
ax2.set_title('Refrigeración')
ax2.legend(loc = 'upper right')

plt.show()
```

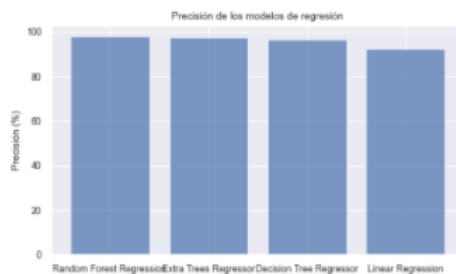


<Figure size 360x360 with 0 Axes>

<Figure size 720x720 with 0 Axes>

Evaluación de los modelos

```
In [75]: # visualización de Las precisiones del algoritmo utilizando Matplotlib
acc = ( 'Random Forest Regression', 'Extra Trees Regressor', 'Decision Tree Regressor' , 'Linear Regression')
x_pos = np.arange(len(acc))
accuracy = [acc_rfr, acc_etr, acc_dtr, acc_linreg]
plt.figure(figsize = (7,4))
plt.bar(x_pos, accuracy, alpha=0.7, align='center', color='b')
plt.xticks(x_pos, acc)
plt.ylabel('Precisión (%)')
plt.title('Precisión de los modelos de regresión')
plt.show()
```



Tanto la regresión utilizando el modelo de Random forest , el modelo de Decision Tree Regressor y el modelo de regresión lineal son modelos aplicables a la predicción del valor de la carga de calefacción y refrigeración. La buena capacidad de predicción de los tres modelos es evidente por el mayor valor de las precisiones (las tres precisiones son >90%). La regresión con el modelo de Random forest parece dar una mejor precisión que las otras dos.