Tetuan City Power Cosumption

Uri Jared Gopar Morales, ITC A01709413, Tecnológico de Monterrey Campus Querétaro.

***Abstracto*- En el siguiente documento se presentará el análisis de los datos sobre el consumo de energía en la ciudad de Tetuán en Marruecos por medio de un dataset.**

I. INTRODUCCIÓN

Tetuán se encuentra en el norte de Marruecos a orillas del Mar Mediterráneo. Esta ciudad se caracteriza por su rica cultura, fue reconocida como Patrimonio de la Humanidad por la UNESCO. Sus principales fuentes de ingresos son la agricultura, pesca, turismo y el comercio. A lo largo del tiempo su población fue aumentando drásticamente tanto por españoles como extranjeros.

1. Evolución de la población por Distrito y Barrio

	Población a 1 de enero										
	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021
06. Tetuán	156.433	155.764	155.684	152.523	150.860	152.545	153.789	155.967	157.937	161.313	159.849
061. Bellas Vistas	29.590	29.262	29.074	28.326	28.074	28.270	28.359	28.750	29.146	29.895	29.465
062. Cuatro Caminos	34.473	34.244	34.252	33.633	33.336	33.637	33.918	34.254	34.608	35.192	34.817
063. Castillejos	20.502	20.343	20.397	19.963	19.639	19.799	19.982	20.363	20.513	20.835	20.541
064. Almenara	22.137	22.179	22.110	21.897	21.651	21.769	21.982	22.421	22.642	23.003	22.738
065. Valdeacederas	25.231	25.286	25.494	24.921	24.583	24.932	25.214	25.571	26.021	26.545	26.625
066. Berruguete	24.500	24.450	24.357	23.783	23.637	24.198	24.314	24.608	25.007	25.843	25.663
Ciudad de Madrid	3.269.861	3.237.937	3.215.633	3.166.130	3.141.991	3.165.883	3.182.175	3.221.824	3.266.126	3.334.730	3.312.310

2. Números índice por Distrito y Barrio. Base 2010 = 100

	Población a 1 de enero										
	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021
06. Tetuán	100	99,6	99,5	97,5	96,4	97,5	98,3	99,7	101,0	103,1	102,2
061. Bellas Vistas	100	98,9	98,3	95,7	94,7	95,3	95,8	97,2	98,5	101,0	99,6
062. Cuatro Caminos	100	99,3	99,4	97,6	96,7	97,6	98,4	99,4	100,4	102,1	101,0
063. Castillejos	100	99,2	99,5	97,4	95,8	96,6	97,5	99,3	100,1	101,6	100,2
064. Almenara	100	100,2	99,9	98,9	97,8	98,3	99,3	101,3	102,3	103,9	102,7
065. Valdeacederas	100	100,2	101,0	98,8	97,4	98,8	99,9	101,3	103,1	105,2	105,5
066. Berruguete	100	99,8	99,4	97,1	96,5	98,8	99,2	100,4	102,1	105,5	104,7
Ciudad de Madrid	100	99,0	98,3	96,8	96,1	96,8	97,3	98,5	99,9	102,0	101,3

Causando como resultado un aumento en la demanda energética, esto afecto directamente a los sectores clave como la industria, comercio y el medio ambiente.

Como precaución se registraron los datos del consumo de energía del año 2017, se midieron diferentes atributos los cuales fueron, la fecha con registros cada 10 minutos, temperatura, humedad, velocidad del viento, flujos difusos generales y flujos difusos, se dividió al territorio en 3 zonas diferentes para poder saber el consumo de energía de estas.

II. MANEJO DE DATOS

El dataset cuenta con 52,417 por cada día del año que se tomaron las medidas, para tener una vista panorámica de las condiciones en las cuales el consumo de energía es mayor en las zonas.

Sin embargo, nos enfocaremos en predecir el consumo de la primera zona y los meses de junio, julio y agosto de 2017. Dejándonos con 13,392 registros. Se decidió la zona 1 por su principal consumo de energía a través del año, ya que es mayor que el de las zonas 2 y 3. Al seleccionar los meses fueron los que más consumo tuvieron durante ese año, nos ayudaran a poder predecir el consumo máximo de los siguientes años, únicamente utilizaremos 3 meses para tener un mejor resultado de las predicciones mejor sin muchos sesgos por aquellos meses que el consumo de energía fue menor, pero que otras variables afecto dicho consumo.

```
Los dos meses con mayor consumo en Zone
Mes: 8, Consumo: 162646686.26 unidades
Mes: 7, Consumo: 159952055.28 unidades
Mes: 6, Consumo: 149495936.42 unidades
```

Teniendo definido nuestra área de estudio, se realizó los ajustes al dataset, los cuales fueron la normalización de las variables a estudiar, para que todos los valores se encuentren en un rango de [0 a 1], facilitando de esta manera el análisis y dejando los posibles errores de que las variables más grandes sesguen a las pequeñas causando esto discrepancia en los datos, la fórmula que utilice para dicha actividad fue la de normalización Min-Max:

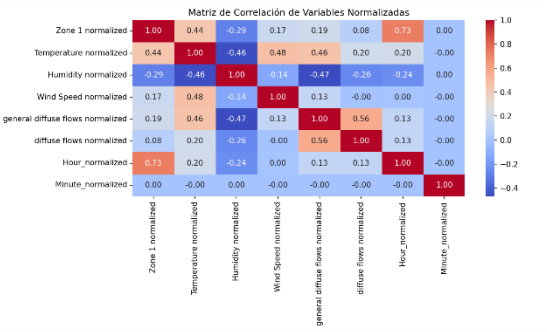
$$Xnormalized = \frac{x - x_{min}}{x_{max} - x_{min}} = \epsilon\ [0,1]$$

Teniendo los valores normalizados, ahora vamos a estandarizar dichos datos, esta función nos va a ayudar con nuestra distribución de datos los cuales los va a centralizar.

$$Xstandardized = \frac{x - mean(x)}{std(x)}$$

III. MODELO

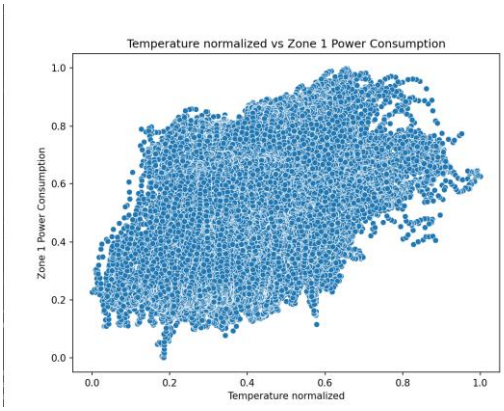
Como punto de partida seleccione la regresión lineal múltiple, para lograr la predicción del consumo de energía en la zona 1 de la ciudad de Tetuán. Para comprender las diferentes variables de estudio con su comportamiento gráfique mi matriz de Correlación de Variables Normalizadas, es de gran ayuda para visualizar de una forma más clara como las diferentes variables tienen un efecto en nuestra variable dependiente que sería la zona 1, por lo que si el numero de la matriz es 1 indica una correlación positiva lo que hace que si esa variable sube nuestra variable dependiente también lo hará.



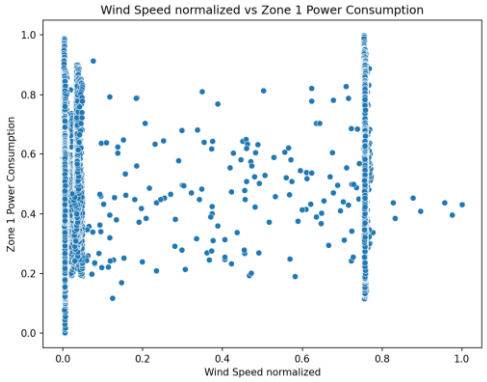
Como un resumen rápido de esta matriz, podemos observar que las variables que más influyen en el consumo de energía son: las horas, la temperatura, velocidad del viento y los flujos difusos generales. Obteniendo esto ya sabemos que variables nos ayudaran el correcto entrenamiento de nuestro modelo para una mejor predicción. Las demás variables no afectan mucho.

Vamos a ver un poco como es el comportamiento de los datos y su influencia en el consumo de energía.

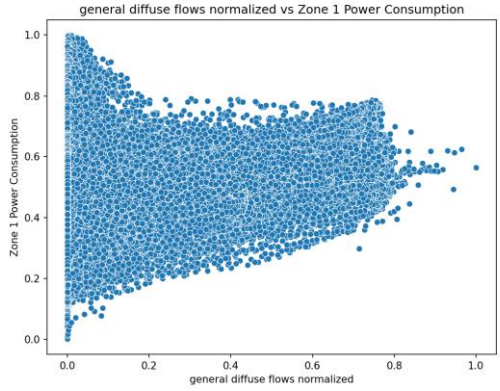
Relación entre temperatura y zona 1.



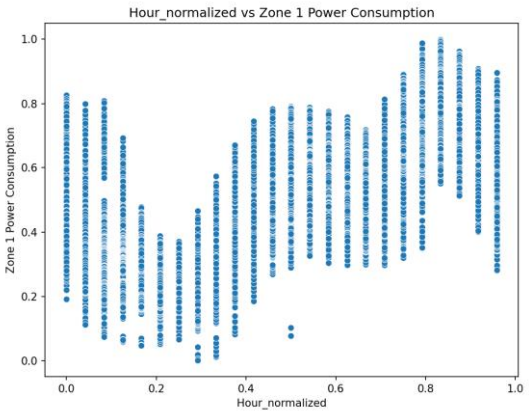
Relación entre velocidad del viento y zona 1



Relación de flujos difusos generales y zona 1



Relación hora y zona 1



Estas gráficas me ayudaron a poder comprender que el comportamiento de la dispersión no es perfectamente lineal, sin embargo, me percate como es que todas las variables tienen influencias en la variable dependiente que en este caso es el total de consumo de energía de la zona 1. Dicho esto, el modelo de regresión lineal como punto de partida me ayudará para poder predecir el consumo de energía, pero al ser este muy simple tendrá algunas limitaciones.

IV. HIPOTESIS

En dicho modelo la hipótesis se utilizará para predecir los valores del consumo de energía de una muestra dado una entrada. La función de esta hipótesis luce de esta manera:

$$h_\theta = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

Por el momento esta es la mejor función, esto se debe a que al tratar de predecir valores continuos como es el consumo de energía, este no contiene la limitante del rango como lo tiene la función sigmoide, por lo que la salida puede tomar todos los números reales y no limitarse entre 0,1.

V. FUNCIÓN DE COSTO

Tiene como propósito enseñar la diferencia que existe entre las predicciones y los valores reales, al minimizar dicha función por iteración lo que se genera es que ajustamos los parámetros para mejorar las predicciones del modelo.

$$J_\theta = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$$

Lo que realizamos es poder minimizar el Error Cuadrático Medio, es crucial para modelos de regresión porque proporciona una media clara y continua de cuán bien el modelo esta realizado con respecto a los datos disponibles, guiando los parámetros de optimización.

VI. DESCENSO GRADIENTE

Nos ayuda a que el modelo aprenda de los datos minimizando la función de costo. Luego que calculamos la gradiente de la función de costo respecto a los parámetros por primera vez, lo que hace es que los parámetros se van a ir actualizando en la dirección que reduce el costo y aumentan las predicciones.

VII. ENTRENAMIENTO

Para implementar esta regresión lineal, separe mi dataset en 60% entrenamiento, 20% valores de prueba y 20% validación, con la finalidad de poder ajustar los parámetros al modelo y de esta manera poder aprender las relaciones entre las variables independientes y la variable dependiente.

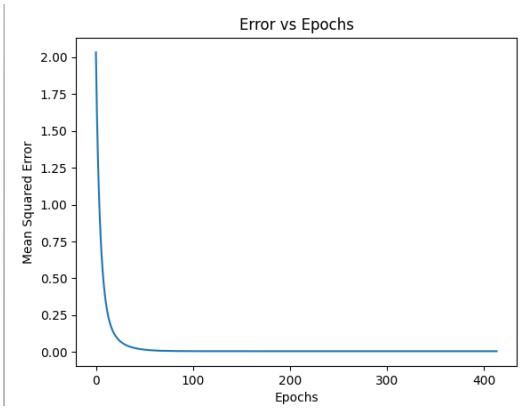
- Entrenamiento 60%: nos ayuda para entrenar el modelo, durante este proceso ajusta sus parámetros para minimizar sus predicciones.
- Validación 20%: Evalúa al modelo durante su entrenamiento, este ajusta los hiperparámetros como la tasa de aprendizaje con datos que nunca ha visto
- Prueba 20%: Se utiliza para evaluar el rendimiento final del modelo, este nos ayuda para ver qué tan acertado es

nuestro modelo con las predicciones de datos que nunca ha visto.

Para implementar esto hice uso del algoritmo de gradiente descendiente, para minimizar el error en las predicciones a lo largo de las épocas. Tras terminar dicho entrenamiento el modelo se prueba con datos no vistos para evaluar su capacidad de generalización. La precisión final demostrara la capacidad que tiene el modelo para predecir escenarios en la vida real.

VIII. RESULTADOS

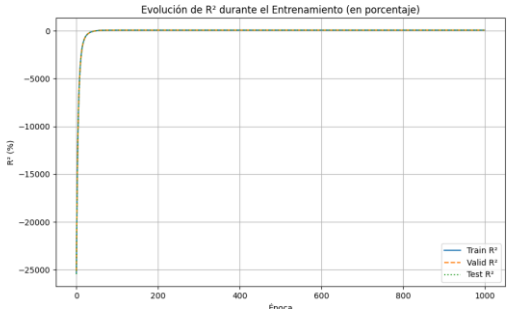
Los resultados obtenidos a través de dicho modelo fueron del 0.6318 lo que nos indica que el modelo explica el 63% de la variabilidad en los datos de prueba en el consumo de energía.



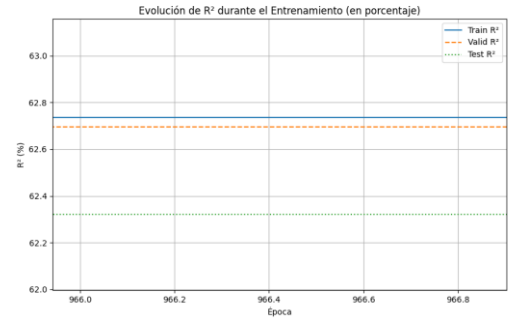
Aunque es un poco baja tiene sentido, ya que al ser esta una regresión lineal no puede predecir los valores más altos o bajos.

IX. GRADO DE BIAS O SESGO

Para poder saber como fue la evolución del modelo durante las épocas así de como fue aprendiendo realice la siguiente gráfica.



En esta se puede apreciar como el modelo fue aprendiendo, parece que solo es una línea pero que pasa si la vemos más de cerca:




```
Epoch 600, Train R²: 62.74%, Valid R²: 62.70%, Test R²: 62.32%
Epoch 700, Train R²: 62.74%, Valid R²: 62.70%, Test R²: 62.32%
Epoch 800, Train R²: 62.74%, Valid R²: 62.70%, Test R²: 62.32%
Epoch 900, Train R²: 62.74%, Valid R²: 62.70%, Test R²: 62.32%
```

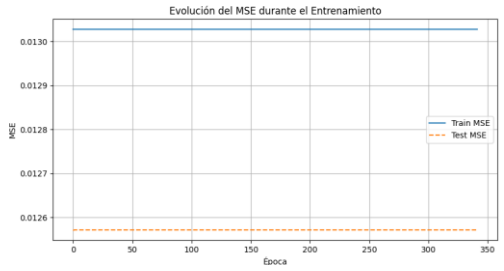
Podemos notar que las R^2 del conjunto de entrenamiento, validación y prueba su porcentaje de explicación de variabilidad de los datos se quedo muy abajo y se mantuvieron constantes, lo que nos indica que el modelo no está aprendiendo lo suficiente.

También podemos observar que a lo largo de las épocas los valores siguen permaneciendo sin mejoras significativas.

Estos resultados nos dicen que el modelo cuenta con un problema de underfitting, se debe a que el modelo al ser demasiado simple no este capturando suficiente información de los datos para realizar las predicciones precisas. Para reducir un poco el impacto de esto underfitting reduje el valor de regularización, ya que al ser muy alto reduce la capacidad para ajustarse adecuadamente a los datos.

X. VARIANZA

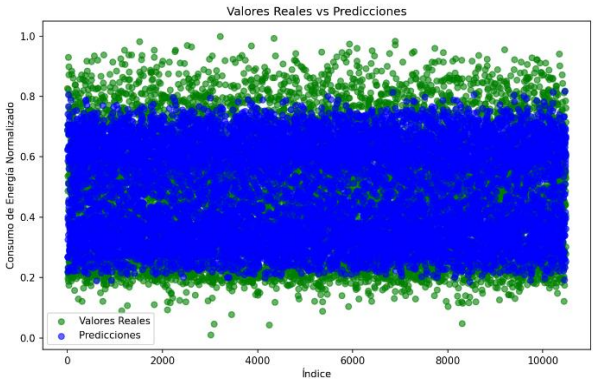
La varianza se observa igualmente en el MSE para los conjuntos de entrenamiento y prueba. Al tener ambos valores similares y bajos, indica que el modelo tiene una varianza baja, indicando que el modelo tampoco sufre de overfitting significativamente a los datos de entrenamiento. Esto se puede apreciar en el siguiente gráfico, al ser las gráficas paralelas demuestra que el modelo converge rápidamente y no necesita muchas iteraciones para alcanzar su mejor rendimiento, lo que ocasiona que no esté capturando ruido de los datos de entrenamiento.



XI. CONCLUSIÓN

A pesar de que el ajuste general es aceptable, debemos tomar en cuenta la distribución del dataset, ya que, si algunas características son más comunes, el modelo se puede ir sesgando hacia estos patrones más frecuentes.

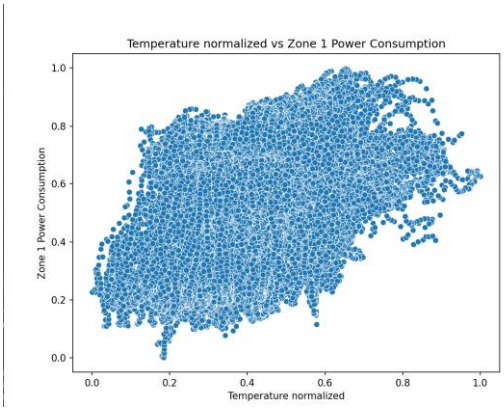
El modelo actual podría no ser suficientemente complejo para capturar todas las relaciones no lineales o interacciones entre características, lo que podría explicar por qué no se alcanza un R^2 más alto. En el siguiente grafico se muestra más claro los valores de predicción vs los reales del modelo.



XII. RED NEURONAL

Para poder mejora las predicciones del consumo de energía en la zona 1, realice un modelo más robusto el cuál es la red neuronal.

Este consiste en realizar capas de neuronas (nodos) que procesan y transforman los datos para que logren aprender patrones complejos como los que tiene el dataset, un ejemplo de estos patrones sería la relación de temperatura y de energía.



A. IMPLEMENTACIÓN

Para realizar dicha red neuronal use el framework de PyTorch, este me permite realizar redes neuronales flexibles y realizar cálculos automáticos.

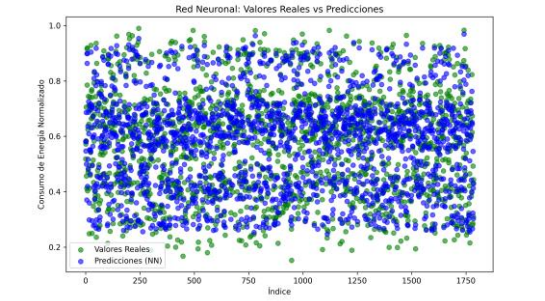
La red neuronal cuenta con una capa de entrada, en la cual se le pasaran todos los datos iniciales: hora normalizada, temperatura, humedad, etc.

En las capas ocultas: Es donde se lleva a cabo el procesamiento de estas entradas aplicando transformaciones no lineales (función de activación ReLU). Mientras más neuronas tengamos el permite que el procesamiento sea más abstracto y complejo, yo decidí únicamente por 128,64,32 en estas capas ocultas.

Capa de salida: Tiene una sola neurona que predice el consumo de energía normalizado.

Decidí solo tener 3 capas ocultas para evitar el overfitting, ya que en este modelo puede ocurrir cuando el modelo se vuelve tan complejo que aprende no solo los patrones de los datos, también el ruido que existe en ellos.

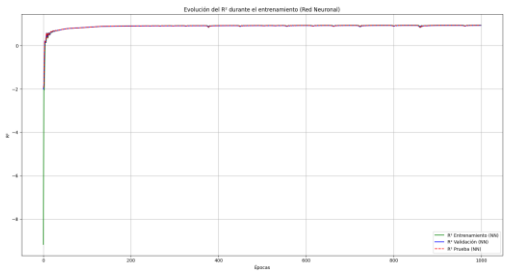
B. RESULTADOS



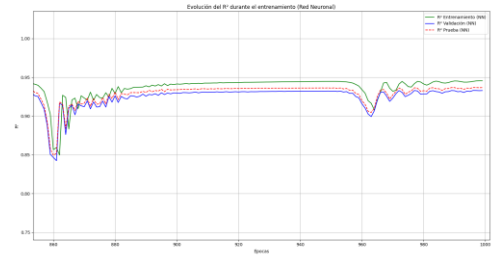

```
Resultados del modelo de Red Neuronal:
MSE en el conjunto de entrenamiento: 0.0020
MSE en el conjunto de prueba: 0.0021
R² en el conjunto de entrenamiento: 0.9405
R² en el conjunto de prueba: 0.9369
```

Como podemos observar las predicciones aumentaron sus R^2 al 93% un 30% de acurrency que al realizar con el modelo de regresión lineal.

Vamos a ver como fue la evolución del modelo a través de las épocas.



Vamos a acercarnos más:



Aquí podemos apreciar cómo es que, si fue aprendiendo poco a poco a diferencia de nuestro modelo de regresión lineal, ya que no se muestran con datos constantes o que se quedaron en atascados en los datos.

```
Epoch 700/1000, Train Loss: 0.0020, Train R²: 0.9394, Val R²: 0.9284, Test R²: 0.9325
Epoch 800/1000, Train Loss: 0.0026, Train R²: 0.9220, Val R²: 0.9025, Test R²: 0.9076
Epoch 900/1000, Train Loss: 0.0020, Train R²: 0.9406, Val R²: 0.9299, Test R²: 0.9339
Epoch 1000/1000, Train Loss: 0.0018, Train R²: 0.9455, Val R²: 0.9329, Test R²: 0.9367
```

Lo que podemos apreciar es que el modelo esta capturando bien los patrones generales sin general overfitting, al no aprender de memoria los valores de entrenamiento.

XIII. RANDOM FOREST

Es un algoritmo que se basa en la construcción de múltiples arboles de decisión. Es un modelo robusto que maneja los datos con patrones no lineales, es muy útil para los datos que tienden a tener mucho ruido, la ventaja que tiene este contra la red neuronal es que es más fácil de ajustar y no requiere tanto procesamiento.

A. IMPLEMETNACIÓN

Utilize el framework de Scikit-learn.

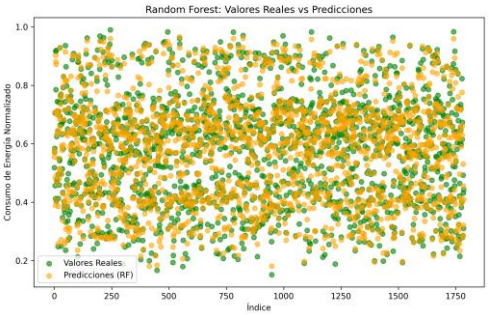
Para construir mi Random Forest configure el modelo con 200 árboles de decisión, para que realicen diferentes predicciones.

Esto ocurre porque cada árbol se entrena dividiendo los datos en grupos más pequeños.

En cada nodo del árbol el modelo selecciona una característica como la hora, temperatura, humedad, etc. Esto con el propósito que los grupos sean más homogéneos.

Una vez que todos han sido entrenados realizan una predicción para los datos prueba, lo que hace es promediar todas las predicciones obtenidas para obtener el resultado final.

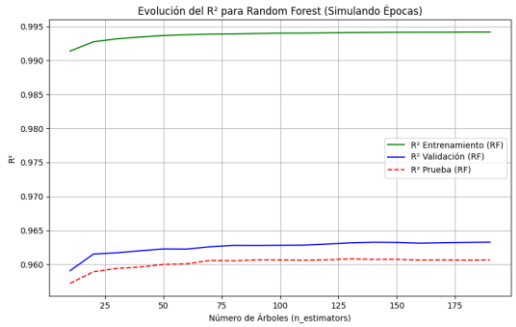
B. REUSLTADOS




```
Resultados del modelo Random Forest:
MSE en el conjunto de entrenamiento: 0.0002
MSE en el conjunto de prueba: 0.0011
R² en el conjunto de entrenamiento: 0.9951
R² en el conjunto de prueba: 0.9689
```

Vamos a ver como fue el aprendizaje del Random Forest. Cabe destacar que esta gráfica luce diferente a las otras por la naturaleza de este modelo, este modelo no aprende por medio de épocas, sino por número el número de árboles.

Entonces cada que el modelo entrena un número específico de árboles, quiere decir que no es una curva continua, porque no existe el ajuste progresivo a diferencia de la red neuronal, por lo que cada punto representa el rendimiento después de agregar más árboles.



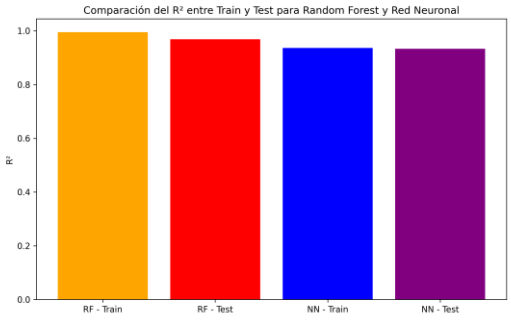

```
Número de árboles: 170, Train R²: 0.9942, Val R²: 0.9632, Test R²: 0.9607
Número de árboles: 180, Train R²: 0.9942, Val R²: 0.9632, Test R²: 0.9606
Número de árboles: 190, Train R²: 0.9942, Val R²: 0.9633, Test R²: 0.9607
```

Pero como podemos apreciar en los resultados obtenidos el Random Forest con 200 árboles obtuvo un acurrancy del 96% siendo mejor que la red neuronal y la regresión lineal.

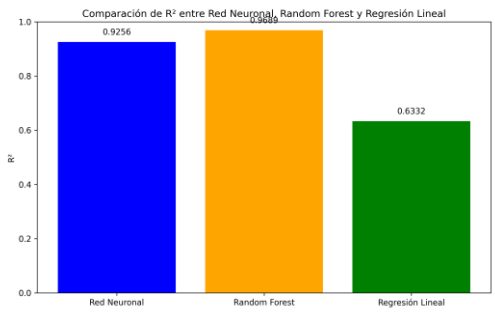
CONCLUSIÓN

El Random Forest predice mejor porque es un modelo más robusto con conjuntos de datos con variables limitadas y sin demasiada complejidad no lineal. La red neuronal, aunque es más poderosa, requieren mayor fine-tuning para lograr un desempeño superior, y en un conjunto de datos como este, es posible que no haya suficiente complejidad para que la red neuronal pueda superar a Random Forest sin un ajuste más preciso.

Se puede apreciar en estas gráficas la diferencia entre los datos de entrenamiento y de prueba de la red neuronal con el Random Forest



Para poder comprender la diferencia de predicciones de cada modelo de sus R^2 realice el siguiente gráfico.



Como había dicho en el gráfico anteriormente, el modelo de regresión lineal al ser muy simple sufre mucho de underfitting ya que no logra predecir los valores más variados, simplemente predice los que se encuentran más cercanos, causando una predicción muy poco confiable, la red neuronal al ser un modelo más complejo ayuda mucho a la predicción de estos valores no lineales, pero requiere un procesamiento intensivo a diferencia del random forest que fue el que mejor predicción me dio, ya que es fácil de entrenar y sintonizar, y en este tipo de problemas con datos no muy grandes y relaciones no lineales, su capacidad para manejar la variabilidad sin mucho ajuste suele darle una ventaja sobre las redes neuronales. Sin embargo cuando tengamos un dataset mucho más complejo la red neuronal va a sobre salir de todos los demás.

Referencias

Explotación del Padrón Municipal de Habitantes 2021. (2021). *Distrito 06 - Tetuán*. Obtenido de https://www.madrid.es/UnidadesDescentralizadas/UDCEstadistica/Nuevaweb/Publicaciones/Padr%C3%B3n%20Municipal%20de%20Habitantes/2021/dto06.pdf

Pykes, K. (07 de 2024). *DataCamp*. Obtenido de Tutorial de PyTorch: Construir una red neuronal sencilla desde cero: https://www.datacamp.com/es/tutorial/pytorch-tutorial-building-a-simple-neural-network-from-scratch?utm_source=google&utm_medium=paid_search&utm_campaignid=21057859160&utm_adgroupid=157296747497&utm_device=c&utm_keyword=&utm_matchtype=&utm_network=g&utm_a

Salam, A. &. (Diciembre de 2018). *Comparison of Machine Learning Algorithms for the Power Consumption Prediction:-Case Study of Tetouan city“*. Obtenido de https://archive.ics.uci.edu/dataset/849/power+consumption+of+tetouan+city

Scikit learn. (03 de 2024). *Random Forest*. Obtenido de https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html

Universidad Piloto de Columbia. (2020). *estudiarvirtual.unipiloto.edu.co*. Obtenido de https://estudiarvirtual.unipiloto.edu.co/blog/evaluar-modelo-de-regresion-logistico#:~:text=El%20cuadrado%20de%20R%20se,predicci%C3%B3n%20y%20el%20valor%20real.