REDES DE NEURONAS ARTIFICIALES

Práctica 1. PROBLEMA DE REGRESIÓN

Predicción del precio medio de la vivienda en California

Curso 2020-21

Realizado por:

Daniel Romero Ureña NIA:100383331

Alejandro Díaz García NIA:100383181

Índice

Introducción

Esta práctica se centra en afrontar un problema de regresión basado en la predicción del precio medio de la vivienda en diferentes distritos de California a partir de una determinada información. Para llevar esto acabo, se hará uso de diferentes modelos de redes de neuronas artificiales, como es en este caso Adaline y el Perceptrón Multicapa, mediante los cuales se obtendrá un modelo óptimo de predicción tras la realización de numerosas pruebas. Para la implementación de Adaline se hará uso del lenguaje Python, mientras que, para la implementación del Perceptrón Multicapa, se hará uso del lenguaje R.

Finalmente, tras haber experimentado con ambos modelos de redes de neuronas artificiales y haber obtenido sus mejores modelos, se realizará una comparación entre los resultados obtenidos mediante Adaline y el Perceptrón Multicapa concluyendo cuál de ellos resulta más preciso y que pros y contras tiene cada modelo respecto al otro.

Preprocesado de datos

En el apartado del preprocesado de datos es donde se debe de recoger del fichero de datos los precios de las viviendas en california con sus correspondientes columnas, cada una asociada a un parámetro correspondiente. Una vez se realiza la entrada de datos en el programa preprocessingData.py se guardan los datos en una matriz de numpy de 17.000 filas y 9 columnas. Una vez se cuenta con los datos dentro de la matriz, se normalizan los valores para que puedan ser tratados por el algoritmo de Adaline. Para evitar todo tipo de sesgo en los datos a la hora de seleccionarlo en los conjuntos de entrenamiento, validación y test se genera una aleatorización de las filas, permutando entre sí para generar un cierto desorden, todo esto bajo una distribución uniforme. Una vez aleatorizado el orden de las filas, se crean los conjuntos, cada uno de los conjuntos con su porcentaje del total del dataset dado de 17.000 filas: datos de entrenamiento (60%, 10200 filas), validación (20%, 3400 filas) y test (20%, 3400 filas). Por último, se sacan en ficheros los nuevos conjuntos para que puedan ser utilizados por el programa Adaline.py, los cuales tienen los siguientes nombres:

- dataout.txt (dataset original con datos normalizados)
- trainData.txt (datos de entrenamiento normalizados)
- validationData.txt (datos de validación normalizados)
- testData.txt (datos de test normalizados)

Adaline

En la siguiente sección se procede a comentar el desarrollo del algoritmo de Adaline en python, así como los resultados que hemos obtenido. Puede verse la implementación en el archivo Adaline.py.

Explicación del código

El código generado en el fichero Adaline.py se encarga de obtener los datos de los tres ficheros creados por preprocessingData.py y guardarlos en matrices de numpy de entrenamiento, validación y test, también obtenemos como entrada del programa mediante la consola el número de ciclos de entrenamiento como primer parámetro y la razón de aprendizaje como el segundo. Tras esto, inicializamos los pesos y el umbral de forma random mediante una distribución uniforme.

Una vez obtenidos los datos iniciales, se procede al ajuste de pesos y umbral, así como la obtención de los errores MSE y MAE de los tres conjuntos de datos, para ello se hará uso de un bucle que permitirá recorrer las 10200 filas del conjunto de entrenamiento, obteniendo tras cada ciclo un peso y umbral ajustados, los cuales serán usados para obtener los errores MSE y MAE tanto del conjunto de entrenamiento como del conjunto de validación por cada ciclo.
Tras haber finalizado la totalidad de los ciclos introducidos, se obtendrán unos pesos y umbral ajustados finales, los cuales nos servirán para la obtención de los errores MSE y MAE del conjunto test, los cuales se han representado en forma de recta discontinua con su valor fijo a lo largo de la gráfica en el eje y para una representación más clara en el contraste entre la evolución de los errores de los conjuntos de entrenamiento y validación.

En lo referido a los archivos de salida, se generan dos archivos, uno de ellos es el modelo que contiene 8 pesos y el umbral en el último lugar. En el otro archivo de salidas *TESTsalidas* se representa la salida obtenida del conjunto de test en la primera columna y por otro lado en la segunda columna tenemos la salida deseada, esta salida deseada está con decimales debido al proceso de desnormalización de los valores, ya que python trunca los valores decimales en las operaciones. Además de esto, se generan dos ficheros con la evolución de los errores MSE y MAE a lo largo de los ciclos de ejecución para los conjuntos de validación y test.

Análisis de resultados

Para la obtención de resultados y la experimentación con diferentes modelos hasta encontrar el mejor modelo, se ha optado por modificar los diferentes tipos de hiperparametros, como diferentes razones de aprendizaje y número de ciclos que permitan un diferente ritmo de aprendizaje durante el entrenamiento con el conjunto de datos. Todo ello con el objetivo de contrastar los valores de los diferentes errores finales obtenidos para los conjuntos.

ID	Nº de ciclos	Razón de aprendizaje	Error final training	Error final validation	Error final test
ad01	800	0.001	MSE: 0.019983 MAE: 0.103709	MSE: 0.021897 MAE: 0.106846	MSE: 0.021509 MAE: 0.106132
ad02	300	0.025	MSE: 0.020362 MAE: 0.104551	MSE: 0.021740 MAE: 0.108305	MSE: 0.021386 MAE: 0.107757
ad03	200	0.1	MSE: 0.022100 MAE: 0.109679	MSE: 0.021802 MAE: 0.107867	MSE: 0.021467 MAE: 0.107500
ad04	500	0.05	MSE: 0.020902 MAE: 0.106210	MSE: 0.021766 MAE: 0.108306	MSE: 0.021431 MAE: 0.107882

Nota: Los resultados anotados en la tabla han sido aproximados haciendo uso de un total de 6 decimales.

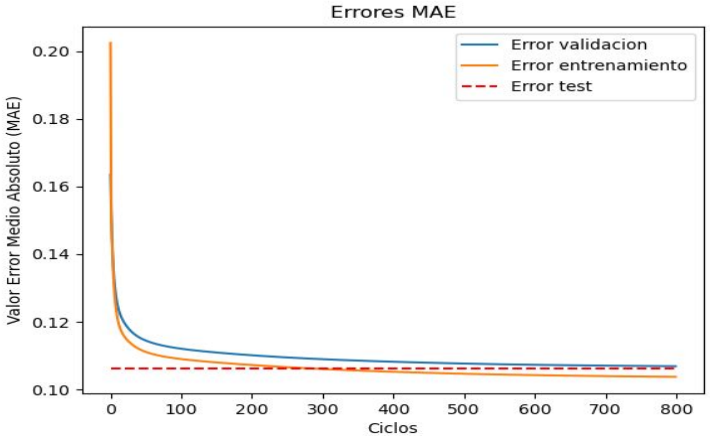
ID	Nº de ciclos	Razón de aprendizaje	Error final training	Error final validation	Error final test
ad05	500	0.025	MSE: 0.020362 MAE: 0.104552	MSE: 0.021740 MAE: 0.108306	MSE: 0.021385 MAE: 0.107757
ad06	300	0.2	MSE: 0.024875 MAE: 0.117100	MSE: 0.022292 MAE: 0.112657	MSE: 0.022144 MAE: 0.112956
ad07	500	0.004	MSE: 0.019945 MAE: 0.103333	MSE: 0.021784 MAE: 0.106475	MSE: 0.021343 MAE: 0.105569
ad08	500	0.15	MSE: 0.023419 MAE: 0.113340	MSE: 0.021903 MAE: 0.109129	MSE: 0.021636 MAE: 0.109039

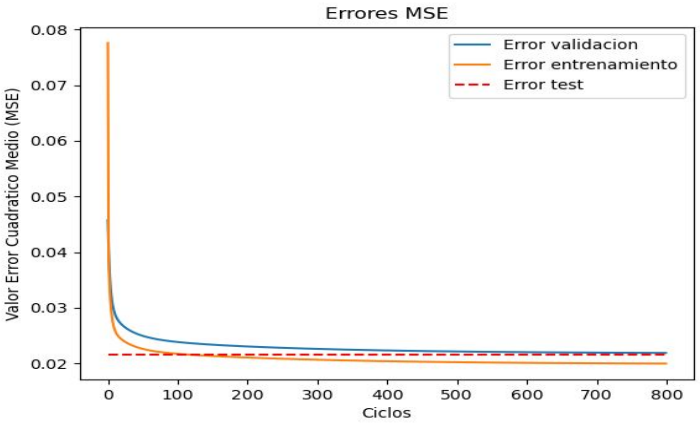
Nota: Los resultados anotados en la tabla han sido aproximados haciendo uso de un total de 6 decimales.

Se puede destacar el interés de los resultados obtenidos por los modelos ad01 y ad06, por lo que se profundizará en ellos, así como la evolución de sus errores a lo largo de los ciclos.

En lo referido al modelo ad01 se puede apreciar que los errores finales MSE y MAE de los conjuntos de entrenamiento y validación se mantienen muy cercanos respecto al valor final de los errores MSE y MAE del conjunto test, siendo este modelo el que menos diferencia de error tiene. Esto es debido a que cuenta con la razón de aprendizaje más baja de los modelos probados, haciendo de este modo que el aprendizaje se realiza de una manera más exhaustiva, lo cual deriva en un mayor tiempo de aprendizaje. De esta manera, este modelo es el más eficaz en la predicción y el más generalizable a distintos conjuntos de datos.

Evolución de los errores a lo largo del tiempo en el modelo ad01:

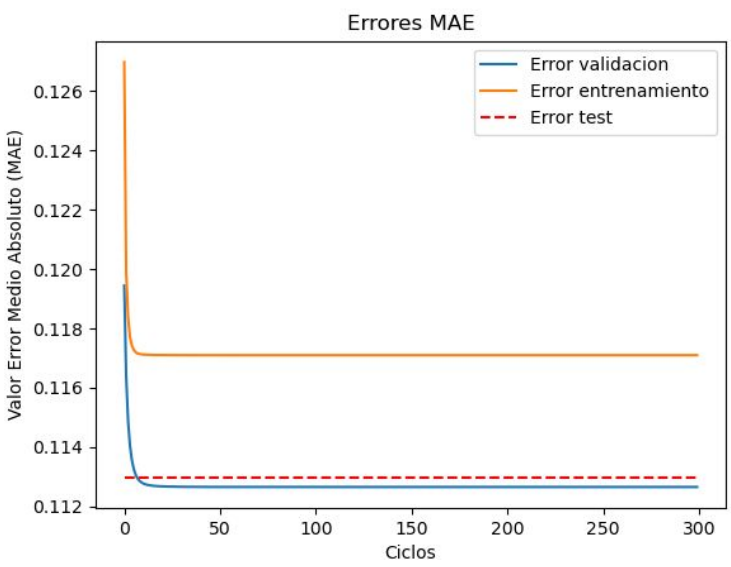


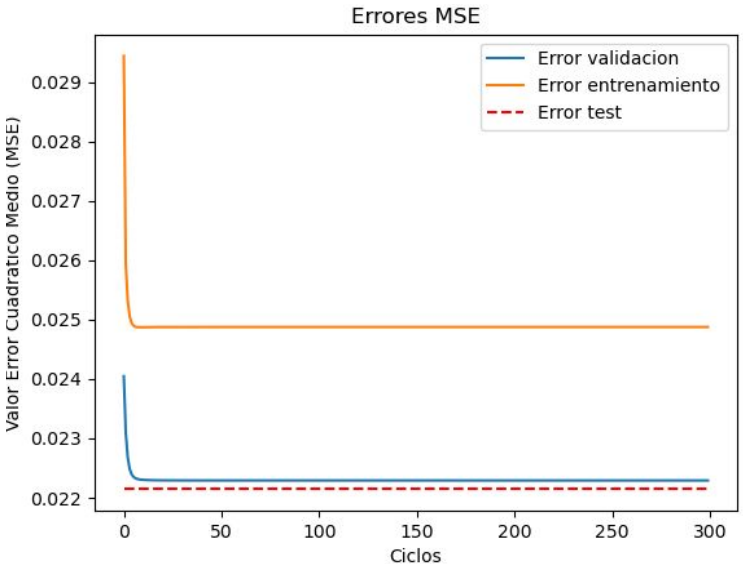


Como se puede apreciar en las gráficas anteriores, la evolución de los errores MSE y MAE a lo largo de los ciclos es muy similar entre el conjunto de entrenamiento y validación, obteniéndose una estabilización de errores distinta cerca del valor de error obtenido mediante el conjunto test, posicionándose el conjunto de entrenamiento como el que cuenta con los mayores errores tras la totalidad de los ciclos y el conjunto de validación como el que cuenta con el mayor.
Además de esto, se puede comprobar la ausencia de overfitting en el modelo, ya que la evolución de los valores de error no pasa a ser creciente en ningún punto de la gráfica pese a hacer uso de un gran número de ciclos.

En cuanto al modelo ad06 podemos observar que los errores de validación y el error de test se encuentran por debajo de los errores de entrenamiento, esto puede darse por la distribución de los datos, pero apenas son diferencias significativas dadas las magnitudes que manejan.

Evolución de los errores a lo largo del tiempo en el modelo ad06:





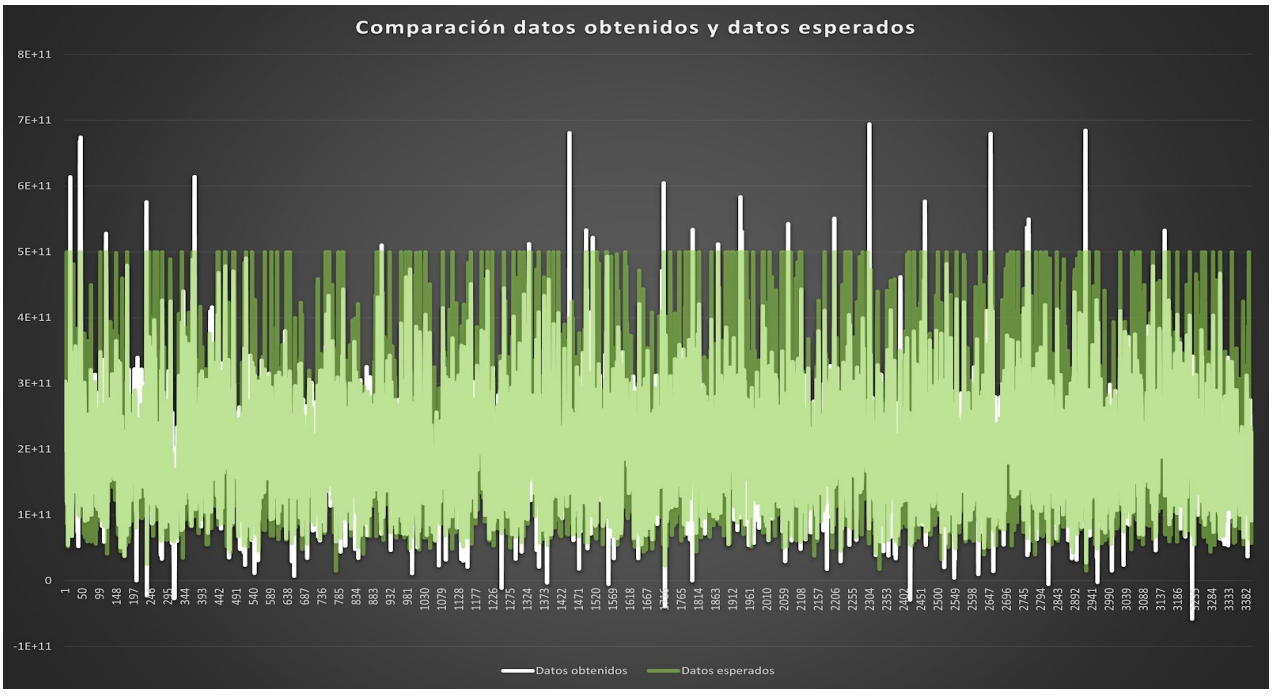
En la gráfica anterior se puede comprobar que pese a existir una diferencia considerable entre los errores obtenidos por el conjunto de entrenamiento y los errores del conjunto de validación, este último obtiene unos valores de error muy cercanos al error de test obtenido, teniendo ambos un bajo valor de error, por lo que a priori demuestra ser eficaz en su predicción. Otro factor curioso de este modelo es el hecho de obtener valores de error superiores o inferiores al error del conjunto test obtenido en función de si se trata del error MSE y MAE.

Se puede concluir que el hiperparámetro clave para la reducción de error en todos los experimentos realizados es el de la razón de aprendizaje, la cual conforme se reduce ocasiona una reducción del error de los tres conjuntos de datos, siendo el hiperparámetro de los ciclos únicamente el que permite la estabilización del error tras diferentes ajustes en los pesos mediante el conjunto de entrenamiento. De este modo, obtenemos como mejores modelos a los que utilizan una menor razón de aprendizaje y un mayor número de ciclos que permite su estabilización hasta su mínimo.

Conclusiones y mejor modelo

Tras realizar pruebas con diferentes tipos de modelos, se puede concluir como modelo más óptimo al modelo ad01, ya que es el que mejor aproxima los valores de error tanto MSE como MAE en los conjuntos de validación y entrenamiento con respecto al conjunto test, contando con valores bajos de error en los diferentes conjuntos, demostrando de este modo ser el modelo más eficaz en cuanto a predicción se refiere de los modelos probados y el más generalizable a conjuntos de datos diferentes con los que se ha entrenado.

Mediante la siguiente gráfica se puede comprobar la diferencia entre las salidas esperadas y obtenidas mediante el modelo ad01, siendo estas salidas pertenecientes al conjunto test:



Nota: Todas las salidas obtenidas cuentan con valores no negativos.

Como se puede apreciar, en términos generales la predicción realizada por el modelo es muy similar a la que se espera, existiendo en algunos casos puntuales una subestimación o sobreestimación, verificando una falta de robustez en la predicción por parte de adaline.

Perceptrón multicapa

En este apartado, se realizarán diferentes pruebas alternando los diferentes hiperparámetros presentes en el perceptrón multicapa para la obtención del mejor modelo de predicción.

Análisis de resultados

Tras experimentar con los diferentes hiperparámetros se han obtenido los siguientes modelos:

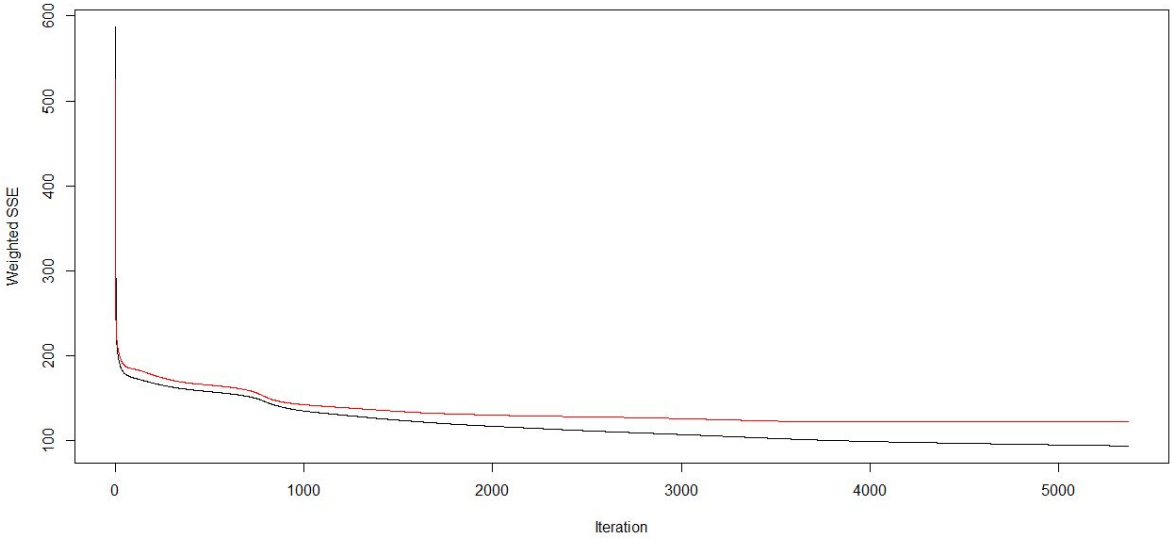
ID	Nº de ciclos	Razón de aprendizaje	Capas ocultas	Neuronas utilizadas	Error final training	Error final validation	Error final test
pm01	9000	0.2	2	50, 50	MSE: 0,008943198593 13339	MSE: 0,011935629996 5062	MSE: 0,012150828336 9031
pm02	5000	0.1	1	33	MSE: 0,012488884429 1762	MSE: 0,013868520885 4604	MSE: 0,014025490937 9408
pm03	9000	0.1	1	80	MSE: 0,011716631549 3303	MSE: 0,013713821502 409	MSE: 0,013728895814 3832
pm04	9000	0.2	1	100	MSE: 0,011134325571 9799	MSE: 0,013219407326 0921	MSE: 0,013305184630 8815
pm05	10000	0.05	1	100	MSE: 0,012301557523 9962	MSE: 0,013753592960 9325	MSE: 0,013971864588 0204
pm06	9000	0.1	1	700	MSE: 0,011616194433 4363	MSE: 0,013526052589 4653	MSE: 0,013459658157 8322
pm07	8000	0.001	1	80	MSE: 0,017446522099 0978	MSE: 0,019043576328 1051	MSE: 0,019034266172 5651
pm08	8000	0.025	2	70,70	MSE: 0,011889708414 0065	MSE: 0,013392123974 5298	MSE: 0,013660901879 2999
pm09	9000	0.05	2	60,80	MSE: 0,010309551547 9896	MSE: 0,012422440618 23	MSE: 0,012543800960 3463
pm10	9000	0.05	2	33,55	MSE: 0,010440398672 3658	MSE: 0,012470710089 1022	MSE: 0,012448655212 3125

En términos generales, se puede apreciar que el uso de dos capas ocultas, a priori, parece tener una mayor eficacia de predicción en comparación con una sola capa oculta, incluso en casos donde se cuenta con una diferencia considerable de neuronas como es el caso del modelo pm06, el cual obtiene peores resultados en contraste con los modelos de dos capas cuya suma de neuronas utilizadas no supera las cien. Además de esto, se ha podido comprobar que un mayor uso de neuronas supone una reducción del valor de error obtenido, dándose esto tanto en el uso de una sola capa oculta (como se puede apreciar en los modelos pm03 y pm06), como en los modelos que hacen uso de dos capas (como se puede observar en los modelos pm09 y pm10).

En lo referido a la razón de aprendizaje, al contrario que adaline, este hiperparámetro deja de ser el factor clave en la reducción del error debido al número de neuronas y capas ocultas utilizadas, las cuales influyen en gran medida en el valor del error. Pudiéndose obtener errores de menor valor en razones de aprendizaje con un valor superior, como es el caso del modelo pm01, el cual obtiene valores más óptimos que modelos cuya función de aprendizaje es igual o inferior a 0.05.

El modelo más destacado de los modelos obtenidos mediante las diferentes pruebas es el pm01, el cual pese hacer uso de un valor de razón elevado como es 0.2 obtiene los menores resultados de error en contraste con el resto de modelos, haciendo uso únicamente de dos capas ocultas que hacen uso de 50 neuronas cada una.

La evolución del error MSE del modelo puede apreciarse en la siguiente gráfica:

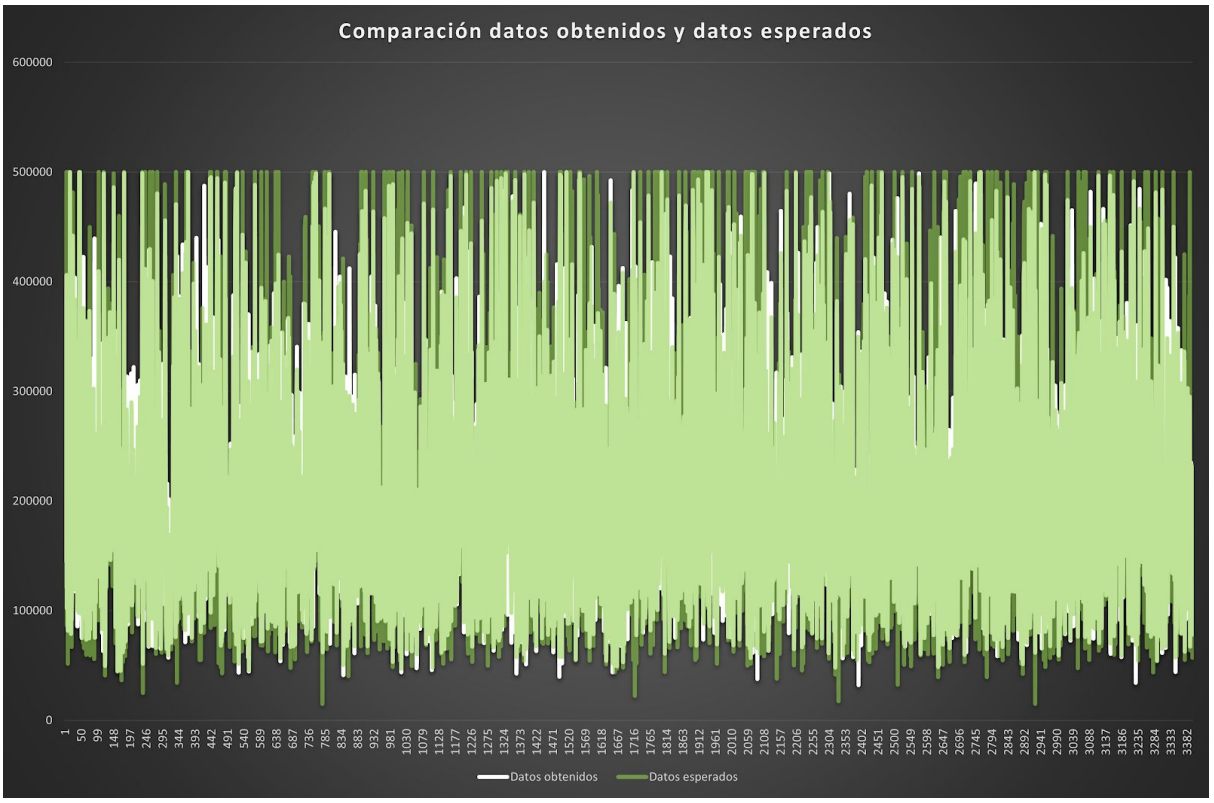


Nota: Los errores MSE de validación son representados en rojo y los de entrenamiento en negro.

Conclusiones y mejor modelo

Tras la experimentación realizada con el perceptrón multicapa, se puede concluir que el modelo más óptimo para realizar la predicción es el modelo pm01, el cual cuenta con el menor error MSE en los conjuntos de entrenamiento, validación y test. De este modo, el modelo pm01 se considera como el modelo más eficaz a la hora de realizar la predicción de salidas de datos, además de ser un modelo generalizable a diferentes conjuntos de datos debido a la similitud en los valores de error MSE entre los conjuntos de validación y test finales.

Mediante la siguiente gráfica se puede comprobar la precisión de predicción del modelo pm001, pudiendo contrastar gráficamente la diferencia entre las salidas predichas por el modelo y las salidas esperadas:



Se puede apreciar una buena precisión en la predicción, obteniendo una predicción cercana al valor esperado en la mayoría de los casos, eliminando la subestimación producida en adaline a los datos con un valor considerablemente mayor respecto al resto de salidas, mostrando de esta forma una predicción más robusta y precisa que la obtenida en adaline.

Comparativa entre Adaline y Perceptrón Multicapa

Tras haber realizado diferentes experimentos con cada tipo de red de neuronas artificial, se ha llegado a una serie de conclusiones sobre cuál de ellas es más conveniente para abarcar el problema planteado con esta práctica. Se ha podido ver en las conclusiones de Adaline que logra una buena predicción de algunas salidas con respecto al valor esperado, pero en otros casos la predicción la realiza erróneamente dando una subestimación o sobreestimación de algunas salidas. Esto se debe a que solo existe una única neurona con un set de pesos para cada entrada, lo que para conjuntos de datos muy grandes lo convierte en una limitación frente al perceptrón multicapa.

El perceptrón multicapa, por su parte, ha mostrado unos resultados bastante satisfactorios teniendo casos en los que la predicción se ajusta de forma casi perfecta a lo esperado y sin tener apenas casos de sobreestimación y subestimación tan destacados como los de Adaline, por lo que se puede concluir que el perceptrón multicapa realiza predicciones más robustas en grandes conjuntos de datos. Este gran éxito radica en su arquitectura, dado que dispone de gran cantidad de neuronas y en algunos casos varias capas de estas, siendo cada neurona capaz de aprender a base de ajustar sus

pesos en base a las entradas, lo cual aumenta su precisión respecto a un modelo con una única neurona como es Adaline.

Por otro lado, cabe destacar que la implementación de Adaline es más simple que la creación de un perceptrón multicapa, lo cual aporta beneficios a la hora de trabajar con el código fuente de la implementación, pero como ya se ha comentado, se obtiene un modelo de RNA con menor eficacia y robustez de predicción.

Los modelos que se han seleccionado en cada uno de los apartados lo han sido debido a su bajo error y cercanía entre el valor final de error del conjunto de validación y test. En Adaline se ha elegido el modelo ad01 y en perceptrón multicapa el pm01, a simple vista para comparar qué modelo sería mejor en la predicción, podría verse mediante el error MSE del conjunto de test, el cual tiene un valor de 0,012150 en pm01 y 0.021546 en ad01. A simple vista, se ve claramente que el error cometido en Adaline es significativamente superior, siendo de casi el doble del obtenido en el perceptrón multicapa, lo cual demuestra la capacidad de aprendizaje mayor con la que cuenta el perceptrón multicapa al usar varias neuronas para un problema de predicción como este.

Como conclusión tras haber expuesto las características y tanto los puntos fuertes como las carencias de ambos tipos de RNA, podemos concluir que el perceptrón multicapa es el tipo de RNA más óptimo para realizar la predicción de este conjunto de datos.

Conclusiones y problemas encontrados

Tras la realización de esta práctica hemos podido reforzar los conocimientos que teníamos sobre las redes de neuronas, los cuales eran en su mayoría teóricos. Hemos podido tener una aproximación práctica a cómo trabajan realmente estos tipos de redes de neuronas, lo que nos ha permitido conocer de primera mano las ventajas y desventajas de ambos modelos. En líneas generales ha sido un trabajo que tanto a nivel práctico como teórico nos ha proporcionado una visión más clara del funcionamiento y las posibles aplicaciones que puede tener este tipo de sistemas a la hora de realizar predicciones sobre conjuntos de datos.

En cuanto a los problemas encontrados en la realización de la práctica no tenemos ninguno que sea destacable, teniendo diversos problemas de menor ámbito principalmente en la implementación de Adaline, concretamente en el manejo de los conjuntos de datos y la implementación ajuste de los pesos a lo largo de los ciclos de ejecución.

A opinión personal ha sido una práctica muy didáctica la cual aparte de interesante nos ha resultado bastante entretenida de realizar por todo lo que ha implicado, desde la elección libre de cómo implementar Adaline hasta usar R, que era un lenguaje que nunca antes habíamos usado. La temática elegida nos ha parecido muy correcta, dado que se trata de un problema real, el predecir cómo van a fluctuar los precios de las viviendas en el estado de California en base a una serie de parámetros.