Instituto Tecnológico y de Estudios Superiores de Monterrey

Implementación de una técnica de aprendizaje máquina sin el uso de un framework

Israel Sánchez Miranda A01378705

Profesor Jorge Adolfo Ramirez Uresti

31 de agosto de 2022

1. **Sobre el modelo.**

Para este momento de retroalimentación se programó un perceptrón con función de activación de signo y un descenso de gradiente estocástico para corrección de errores.

   a. **Descripción del modelo.**

   El código se hizo en Python, se contó con un dataset personalizado con el cual el perceptrón, basado en las características enunciadas en el dataset, clasifica si un organismo es un gato o no. El programa divide el dataset en un set de entrenamiento y un set de pruebas basados en un ratio proporcionado por el usuario, cabe destacar que las filas que el algoritmo escoge como set de pruebas y entrenamiento son completamente aleatorias.

   Posterior a esto el algoritmo comienza el entrenamiento, el vector de pesos (proporcionado por el usuario) es ajustado usando el descenso de gradiente hasta conseguir los resultados deseados de su columna objetivo correspondiente utilizando un learning rate igualmente proporcionado por el usuario.

   Finalmente se procede a probar el vector de pesos final con el set de testeo el cual es un set de datos jamás vistos por el perceptrón, obtener su vector de error, el MSE y la precisión del modelo generado.

   b. **Hiper-parámetros del modelo.**

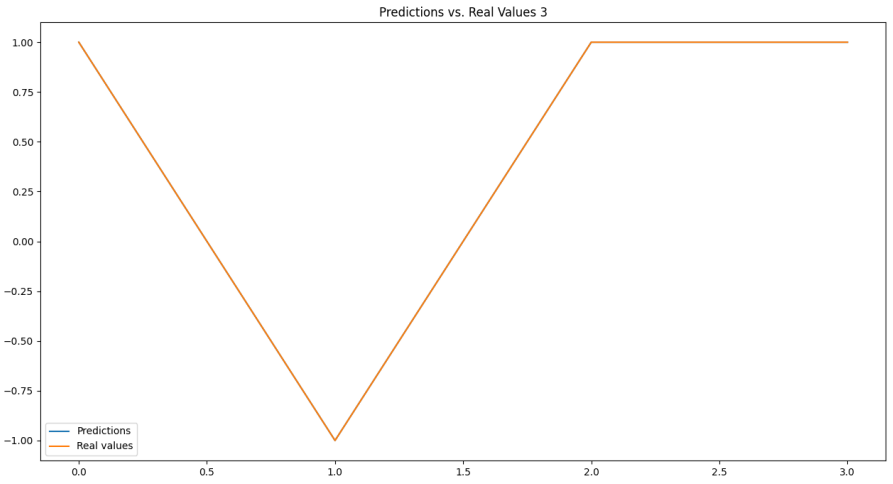
   - `WEIGHT`: Indica el valor que deben de tomar los pesos iniciales del modelo.
   - `ALPHA`: Learning rate del modelo.
   - `RATIO`: Porcentaje del dataset que corresponderá al set de entrenamiento.

2. Pruebas del modelo.

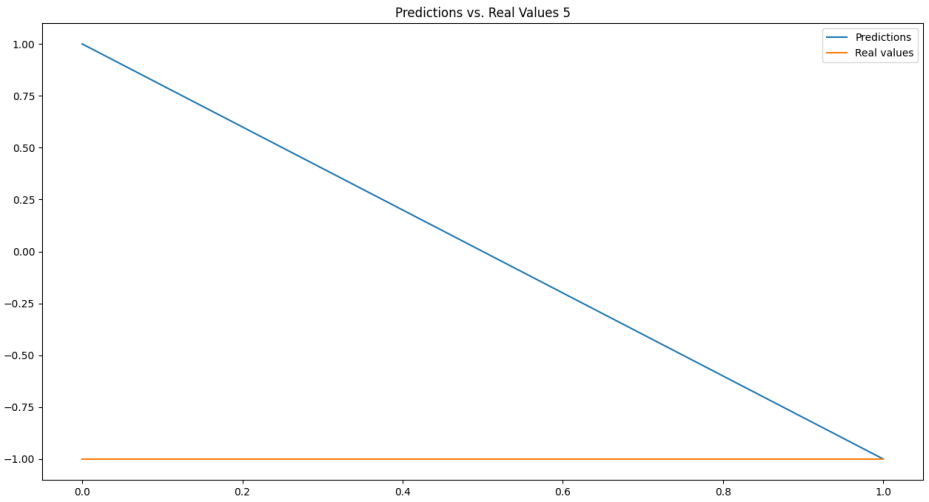
Se realizaron diversas corridas cambiando los valores del vector de pesos, el learning rate y el porcentaje de datos de entrenamiento con la finalidad de ver el desempeño del modelo y ver cómo es que cada uno de estos hiper parámetros afectan su ejecución.

a. Análisis de pruebas.

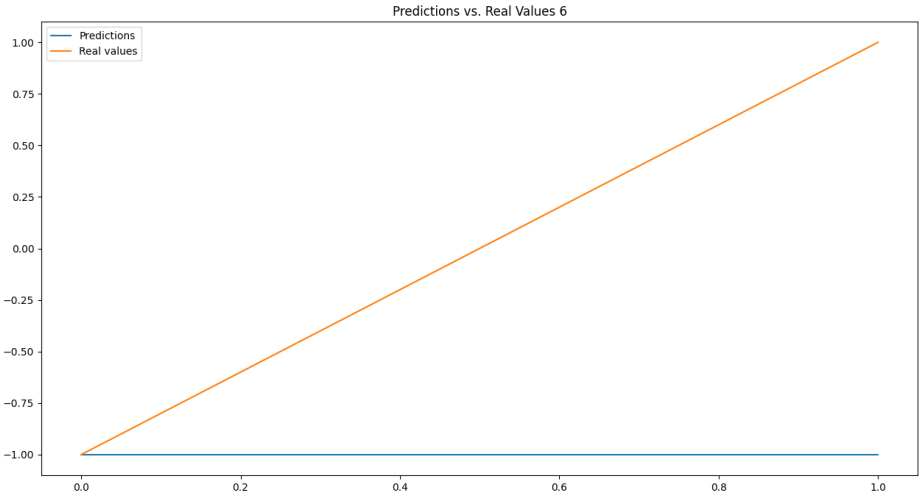
Para este caso se escogieron las 5 corridas más importantes dentro del modelo que describen mejor su desempeño.



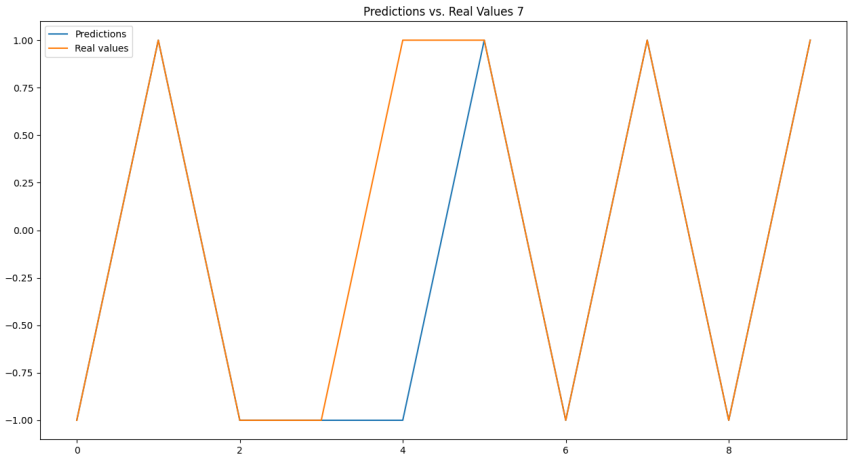
Para este primer ejemplo se usaron valores de pesos de 0.2, un alpha de 0.1 y un 80% del set se dedicó al set de entrenamiento. En este caso se puede apreciar que la precisión del modelo es del 100% y que literalmente no existe error alguno dentro del mismo. Esto puede deberse a que alpha tiene un valor muy pequeño lo que hace que el descenso de gradiente sea más exacto y a que la mayor parte de datos fueron para entrenar. Sin embargo, esto último puede que haya causado un overfitting dentro del modelo.



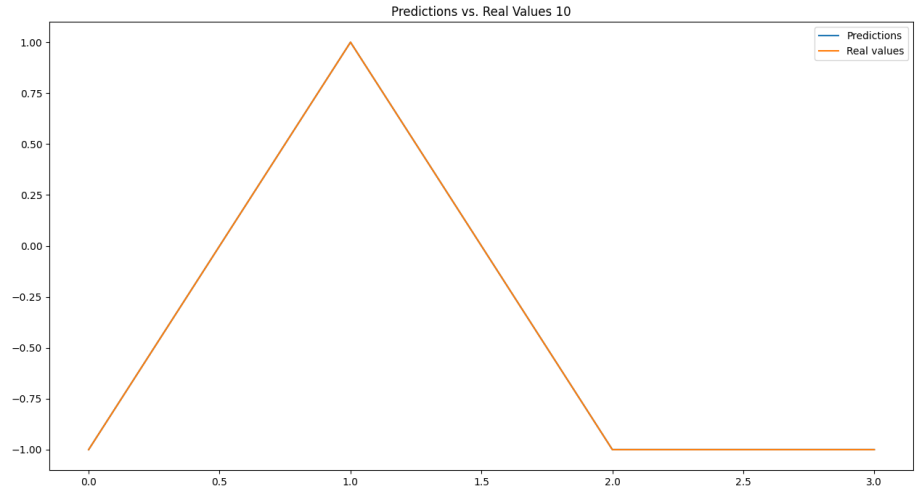
Para este segundo caso se usaron valores de pesos de 0.8, un alpha de 0.9 y un porcentaje de entrenamiento de 90. A pesar de que por este porcentaje es muy probable que el modelo haga overfitting, el alpha tan alto hace que las predicciones sean menos acertadas dando un valor de precisión del 50%.



En este tercer caso los valores son como siguen: pesos iniciales de 0.9, un alpha de 1 y un porcentaje de 90%. Nuevamente se aprecia que el valor de alpha tiene un impacto sumamente grande en la precisión del modelo y que mientras más grande menos precisas serán las predicciones del mismo.



En este cuarto caso se usaron pesos de 0.1, un alpha de 0.1 y un porcentaje de entrenamiento del 50%. En este ejemplo se puede apreciar una vez más que el valor de alpha es muy determinante para la precisión del modelo ya que a pesar de que se tiene la mitad de la muestra para entrenar y la mitad para probar se obtienen resultados muy buenos con solo 2 equivocaciones.



Finalmente, se tiene el último ejemplo que cuenta con los valores más "óptimos" para tener un modelo lo más general y preciso posible: un vector de pesos de 0.1, un alpha de 0.5 y un porcentaje de entrenamiento del 80%. Esto

da un modelo con una precisión perfecta, sin embargo, es importante tomar en cuenta que debido a que los valores que se escogen para el set de entrenamiento y el de pruebas son aleatorios por lo que el resultado podría variar en diferentes corridas.

Gracias a estas pruebas se puede observar que:

- El valor del vector del peso tiene un impacto menor dentro del desempeño del modelo, el valor estándar para este fue 0.1, sin embargo, en pruebas donde se escogió un valor relativamente más alto (0.8-0.9) se obtuvieron predicciones con menor precisión y un MSE un poco más alto.
- El learning rate tiene un impacto muy alto tanto en la precisión del modelo generado como en el tiempo de entrenamiento. Un learning rate pequeño hace que se tenga un modelo relativamente preciso (con riesgo de causar overfitting) pero que haya un tiempo de entrenamiento mayor. Por otro lado un learning rate grande causa menores tiempos de entrenamiento pero un modelo menos preciso, lo cual puede suponer una buena aproximación mientras se mantenga una precisión arriba del 75%.
- El tamaño del set de entrenamiento tiene el mismo efecto que el learning rate pero en menor medida y por motivos diferentes. Si se tiene un 20% de datos de entrenamiento es evidente que se tendrá un entrenamiento menor ya que no se tienen que ajustar tanto los pesos, sin embargo, el resultado tiende a disminuir bastante su precisión. Por otro lado, porcentajes de 80%-90% hacen que se tengan resultados sumamente precisos pero con esto es muy probable que se haga un overfitting del modelo por lo que no es un buen ajuste. Porcentajes entre el 60%-70% hacen que esto sea más general y que sea una mejor

aproximación. Este impacto del porcentaje del set de datos puede deberse a que la muestra es muy pequeña y por eso el tamaño tenga tanto impacto en la precisión y tienda a hacer overfitting cuando el porcentaje es relativamente alto.

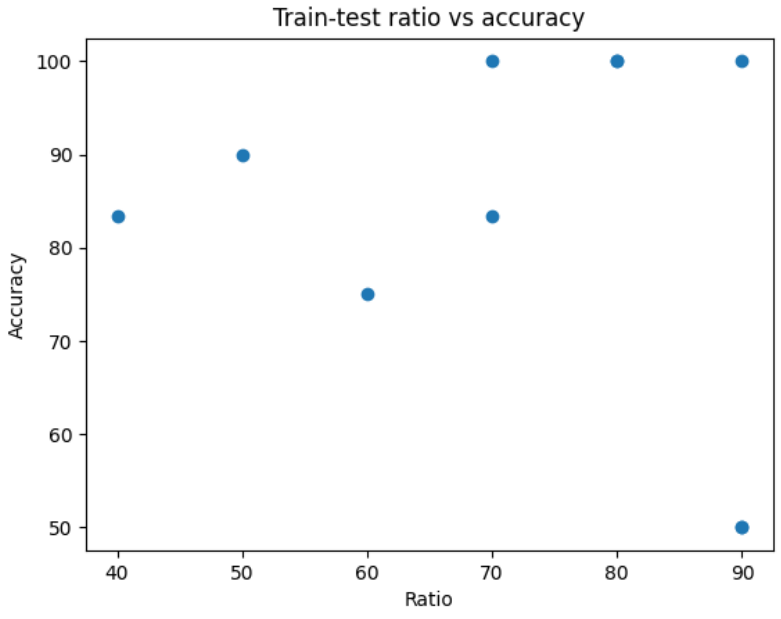
Cabe destacar que la aleatoriedad juega un papel sumamente importante en este caso ya que hubo corridas en donde se tenía un porcentaje de datos de prueba muy alto y se tuvo precisión del 40% o 50%. Esto es bueno ya que se puede apreciar mejor el comportamiento del algoritmo y cómo es que opera con sets de entrenamiento más sesgados hacia -1 o 1.

Las pruebas se pueden apreciar con mayor detalle en los archivos .txt del repositorio.

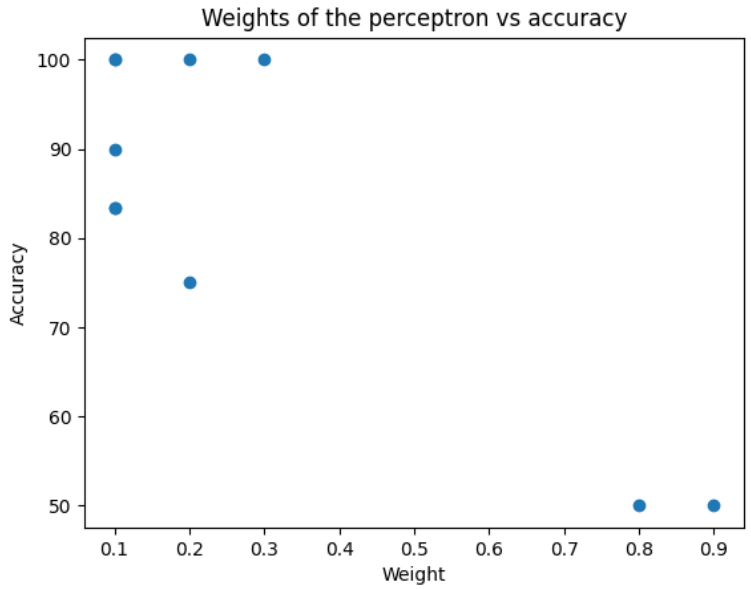
3. Análisis del modelo.

Ahora bien, se muestra el siguiente análisis del modelo para determinar su precisión y si es suficientemente bueno o no.

a. Precisión del modelo.

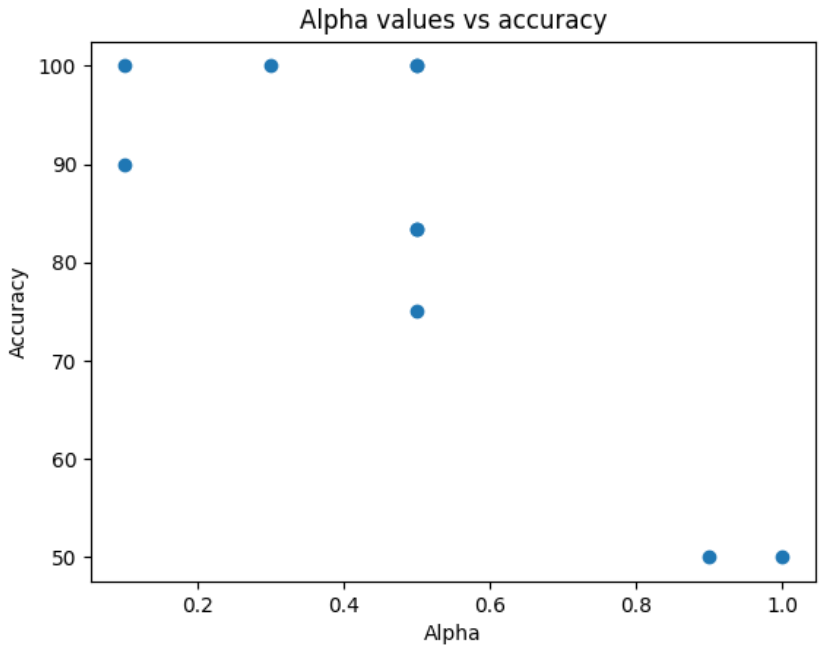


La gráfica anterior compara la precisión del modelo con el porcentaje del set de entrenamiento. En este caso se puede apreciar que, efectivamente, mientras más alto es el porcentaje del set de entrenamiento se tienen resultados más precisos (a excepción del último caso donde con un porcentaje de 90% se tuvo un 50% de precisión). Sin embargo, esto no significa que tener altos porcentajes de entrenamiento sea una buena idea, ya que esto puede tender al overfitting. Por otro lado, también es lógico pensar que al tener más datos para entrenar se podrán clasificar correctamente más registros ya que se cuenta con un set de pruebas más pequeño, por lo que esta gráfica por sí sola no nos puede dar una explicación completa de cómo opera el modelo en su totalidad.

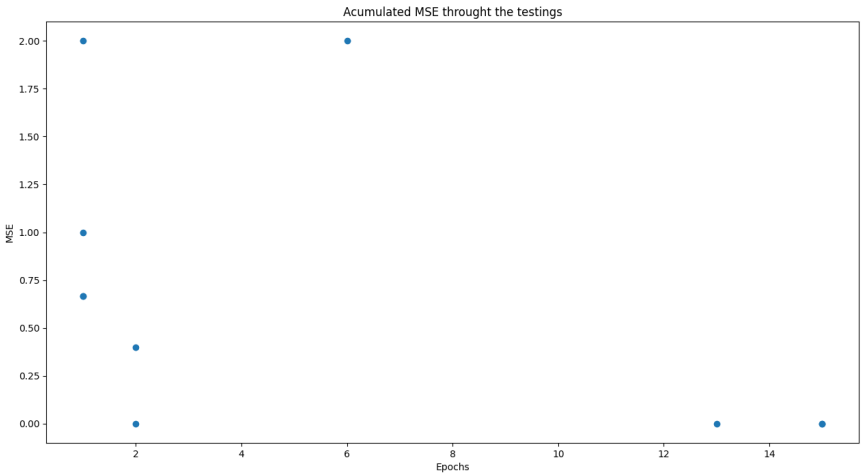


En este caso ocurre lo contrario, mientras más chicos son los pesos del perceptrón más preciso es el modelo, esto dependerá del caso de uso y de la magnitud de los datos que se tengan ya que si se tienen datos con un orden mayor puede que el tener pesos más pequeños haga que el entrenamiento del modelo sea sumamente tardado.

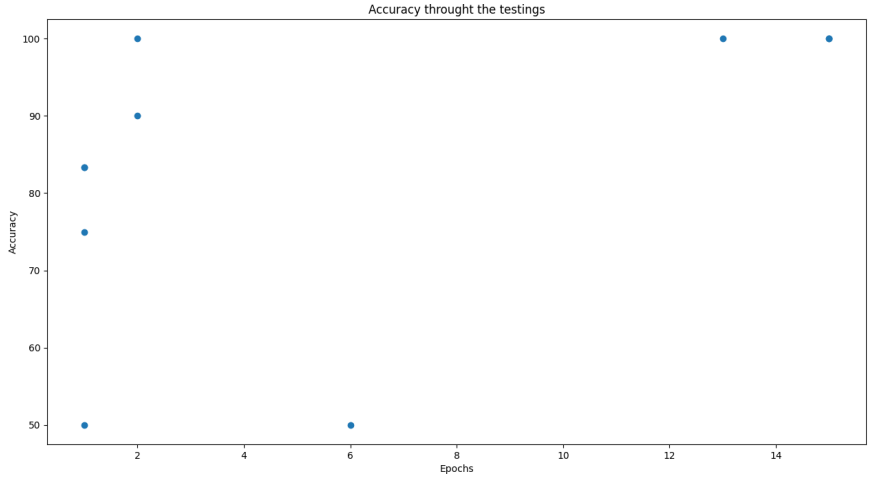
A pesar de esto, para las funciones más generales y comunes del perceptrón, se puede apreciar que con pesos entre 0.1 y 0.3 se obtienen resultados sumamente buenos.



Alpha es el valor más determinante para poder obtener buenos resultados en las predicciones del modelo. Un alpha muy grande da resultados poco precisos ya que el gradiente avanza a pasos poco precisos y sobresaltados. Por otro lado, valores de alpha menores o iguales a 0.5 dan los mejores resultados por un diferencial extremadamente dramático por lo que si se busca generar un modelo con aproximaciones eficientes y acertadas se deberá procurar mantener a alpha en un valor de estas magnitudes.



En esta otra gráfica se puede apreciar que a mayor número de épocas menor MSE se consigue, por lo que, tomando en cuenta que valores de alpha más pequeños obtienen tiempos de entrenamiento (épocas) más largos, es sumamente recomendable escoger valores de alpha pequeños para ir reduciendo el MSE acumulado a lo largo del entrenamiento.



Finalmente, esta gráfica muestra una vez más que a mayor número de épocas mayor será la precisión de los datos debido al ajuste que se le da a los datos. Es importante tomar en cuenta la aleatoriedad de los datos ya que hay casos en los que se tienen pocas épocas y una alta precisión o todo lo contrario. Es por

eso que se recomienda hacer varias corridas con los mismos parámetros para observar el comportamiento del modelo.

b. Evaluación general del modelo.

En general se logró programar de forma exitosa el algoritmo que sigue el perceptrón para reconocer patrones en un set de datos y obtener resultados satisfactorios basados en los hiper-parámetros que el usuario ingrese.

Si se toma en cuenta la aleatoriedad, se escogen valores de alpha pequeños, un porcentaje de entrenamiento de entre el 60% y el 80% y se eligen pesos de acuerdo a la magnitud y caso de uso del problema, el perceptrón programado puede ser una aproximación excelente para obtener predicciones acertadas.