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

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.

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.

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.

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 (0.5-0.6) se obtuvieron predicciones con menor precisión y un MSE mucho más alto.

Por otro lado, el learning rate tiene un impacto 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%.

Ahora bien, el tamaño del set de entrenamiento tiene el mismo efecto que el learning rate pero 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 overfit 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.