Universidad Simón Bolívar
Departamento de Computación y T.I.
Inteligencia Artificial II
Septiembre-Diciembre 2014
Estudiantes: Carnets:
Wilmer Bandres 10-10055
Juan Escalante 10-10227

PERCEPTRÓN Y ADALINE

1. Implemente un perceptrón con n entradas:

- Entrenamiento del perceptrón con una tasa de 0.1, graficando el error ¿Qué concluye?

  Es claro que las funciones AND y OR son linealmente separables, ya que llega un punto en el que el error es cero, siendo ese punto casi inmediato. Por otro lado, la función XOR llega a un punto en el que mantiene siempre un mismo error, que se debe a que no es linealmente separable. Así, cuando el vector de pesos intenta corregirse, lo que se observa es que algunos casos quedan corregidos, y otros casos que estaban clasificados correctamente ahora quedan como casos clasificados incorrectamente. Es justamente por esto que el error se mantiene; oscila de tal forma que nunca puede dividir correctamente los casos de entrenamiento.

- Para otros valores de las tasas de aprendizaje (0.01;0.2;0.5;0.99), ¿Qué conclusiones puede obtener?

  Primero que nada, en esta pregunta se colocó un valor inicial distinto al vector de pesos y se inició cada componente en -3. Esto se debe a que al hacer la inicialización en 0.0 las funciones AND y OR convergían muy rápido y no permitían notar una diferencia sustancial gracias a las tasas. Luego, se observa que dada una tasa más grande, el error decrece con mayor rapidez a medida que crece la iteración.

2. Implemente la regla de entrenamiento delta para una neurona artificial de n entradas (unidad lineal):

- Entrenamiento del adaline con XOR, AND y OR:

  Igual que en el caso del perceptrón, se observa en las gráficas cómo converge el error de las funciones AND y OR, lo que conlleva a una buena clasificación del conjunto de entrenamiento, mientras que en el caso del XOR pasa algo similar al caso anterior, donde éste se queda oscilando con un error de 0.5, que parece ser un óptimo local y no logra obtener una mejor clasificación del conjunto de entrenamiento.

  - Cambiando el valor de las tasas a 0.01, 0.2, 0.5, 0.99 y 0.1, se observa cómo a una mayor tasa la convergencia al óptimo local es mucho más rápida, pero esto solo ocurre en el caso de las tasas 0.01, 0.2 y 0.1, mientras que en el caso del 0.5 el error se mantiene igual para cualquier iteración. Para el caso con tasa de 0.99 el error va incrementando y puede llegar a valores muy grandes; esto sugiere que en el momento del entrenamiento el óptimo local está muy cerca a lo que sugiere el vector de pesos y al hacer un cambio con una tasa muy grande como es la de 0.99, lo que se está logrando es pasar por encima de ese óptimo y sobrepasarlo llegando a otro punto (otro vector de pesos) con un error mucho mayor al anterior.

    Sin embargo, cuando se realiza el experimento con una tasa decremental (o una tasa que decae) se observa cómo la forma en que el error converge a un óptimo local es mucho más lenta (dado que la tasa va siendo cada vez menor), pero lo interesante es que a diferencia del caso anterior, donde la tasa es de 0.99 y el error se dispara a infinito, en este caso el error sí converge a un número pequeño, lo que sugiere que cuando se usan tasas decrementales (con saltos más pequeños) aunque puede converger con mucha más lentitud, se puede lograr la convergencia del error a un número razonablemente pequeño, lo que conlleva a una clasificación bastante buena del conjunto de entrenamiento.

DETALLES DE IMPLEMENTACIÓN Y DE EXPERIMENTACIÓN:

En cuanto a los detalles de implementación, se desarrollaron dos archivos distintos: uno para el perceptrón y uno para el adaline, además de un generador de casos. La forma en que se corre cada uno de estos se encuentra en un “README” que se incluye junto al proyecto.

Por otra parte, se implementó una clase auxiliar llamada mi_vector para facilitar las operaciones de multiplicación de vectores en el caso de la multiplicación de los vectores de pesos con las entradas del conjunto de entrenamiento.

En el caso de la experimentación, se llevaron a cabo pruebas con las tasas sugeridas (de 0.1) y con un vector inicial de pesos que sólo contenía 0's, pero dada la rapidez con la que convergen a las soluciones correctas (tanto el adaline como el perceptrón), se hicieron otros experimentos iniciando cada componente del vector de pesos inicial con un valor de -3, lo que permitió mostrar mejor la diferencia del adaline con el perceptrón con respecto a las demás tasas (0.2; 0.01; 0.5 y 0.99). Los resultados que tienen esta inicialización (de -3 para cada componente) se encuentran bajo la carpeta resultados con el nombre de perceptron_<funcion>_<tasa>_inicio_-3.out, donde función puede ser AND o OR, y la tasa es cualquiera de las tasas pedidas para este ejercicio.

Por último, las gráficas que se encuentran bajo la carpeta de resultados de este proyecto, pertenecen a las funciones AND, OR y XOR, tanto del adaline como del perceptrón con una tasa de 0.1.