Instituto Tecnológico de Costa Rica
Escuela de Ingeniería Electrónica
Profesor: Dr. Pablo Alvarado Moya
EL 5857 Aprendizaje Automático
I Semestre 2024

Entrega: **5 de abril, 2024**

Tarea 4

Softmax y optimización

I. Introducción

En esta tarea se continúa con lo avanzado en la tarea 3. En particular, se explorará la clasificación con softmax, que permite utilizar más de dos clases, y se implementarán otras dos estrategias de descenso de gradiente para el proceso de aprendizaje.

Al igual que la tarea 3, el objetivo principal de esta tarea es implementar los métodos para crear una *intuición* del comportamiento de estos métodos "básicos" multiclase, y los hiperparámetros asociados, antes de revisar métodos más avanzados en el proyecto 1.

Para probar los métodos se utilizará el mismo conjunto de datos de la tarea 3, con tres especies de pingüinos en el archipiélago Palmer en la Antártica.

La invitación al repositorio de trabajo para esta tarea en el Github Classroom está en este enlace, y puede importar allí lo que requiera de su solución de la tarea 3.

II. Procedimiento

II.1. Optimización

En esta sección usted deberá completar la implementación de la clase `optimizer`, ya extendida por usted en la tarea 3.

Su misión concreta en este punto es completar la clase para que realice el proceso de descenso de gradiente utilizando RMSprop ("`rmsprop`") y Adam ("`adam`"). Por tanto, copie el archivo `optimizer.m` con su solución de la tarea 3 a su código de la tarea 4, para continuar su desarrollo.

Utilice el archivo `linreg_main.m`, ya conocido de la tarea 3, como banco de prueba para el proceso de desarrollo de los nuevos métodos de descenso de gradiente solicitados. Allí se realiza una tarea de regresión lineal.

II.2. Softmax

El concepto de softmax es bastante estándar hoy en día en el contexto de redes neuronales y clasificación multi-clase. Dicho concepto de desarrolló en le curso por medio de los modelos lineales generalizados, en la lección 7.

Para este caso, el archivo principal de control de pruebas será `softmax_main.m`.

En esta tarea, al igual que en la anterior, utilizaremos de nuevo la derivación automática para el cálculo del gradiente. Además, emplearemos estrategias de coloreado del espacio de características para interpretar la operación del clasificador.

1. Implemente en el archivo `softmax_hyp.m` la hipótesis $h_{\underline{\boldsymbol{\theta}}}(\underline{\boldsymbol{x}})$ correspondiente a softmax.

   La función de hipótesis debe recibir una matriz de parámetros $\boldsymbol{\Theta}$ y una matriz de diseño $\boldsymbol{X}$ con un dato en cada una de sus m filas.

   Debe asumir que quien llama a esta función ya se encargó de hacer las modificaciones necesarias en los datos para hacer posible el sesgo.

   Particularmente para softmax, es importante que esta función de hipótesis, y la función de pérdida en el siguiente punto, verifiquen por medio de líneas `assert` que las dimensiones de las entradas son consistentes entre sí.

2. Implemente MSE (*mean squared error*) como la función de pérdida (o error) $J(\boldsymbol{\Theta})$ en el archivo `softmax_loss.m` para el caso softmax.

   Esta función recibe como argumentos la matriz de parámetros $\boldsymbol{\Theta}$ (`Theta`), los datos de entrada como matriz de diseño $\boldsymbol{X}$, y las etiquetas de salida $\underline{\boldsymbol{y}}$ codificadas en *one-hot*, con una columna por clase, con un 1 si el dato correspondiente pertenece a la clase representada por esa columna, o 0 si no. Recuerde que en el softmax, $\boldsymbol{\Theta}$ usa una clase menos en la codificación, por estar la probabilidad de la última clases indirectamente definida por todas las otras clases.

3. Pruebe sus funciones agregando código en el archivo `softmax_main.m` para hacer un clasificador que use softmax, usado las cuatro características disponibles como entrada, para estimar a cuál de las tres especies pertenece el espécimen.

   Recuerde modificar los datos disponibles para que el modelo pueda utilizar sesgo. Además, recuerde normalizar los datos, particularmente en este caso que cada característica tiene un rango de valores distinto a los demás.

4. Utilizando las cuatro características disponibles a la entrada, grafique la pérdida con cada método de optimización implementado (incluyendo los de la tarea 3), en función del número de iteración, como se ilustra en la figura 1.

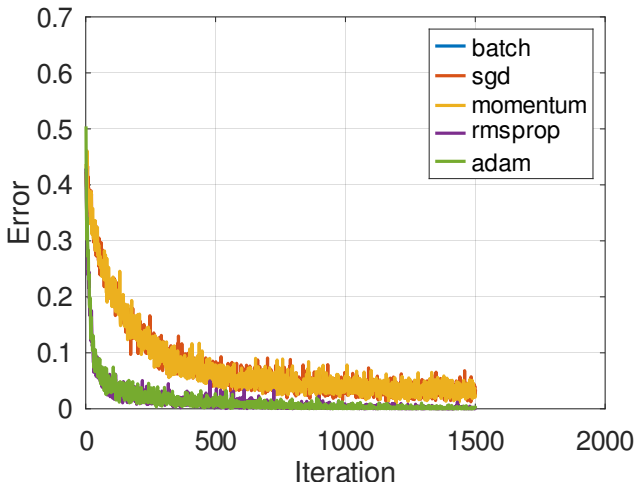

Figura 1: Pérdida en función de la iteración para los cinco métodos de optimización.

Despliegue los valores de $\boldsymbol{\Theta}$ obtenidos en cada caso. Razone qué características parecen ser las más relevantes según esos valores.

5. Implemente una función que calcule el error empírico de clasificación, en este caso, multiclase.

6. Ahora, determine experimentalmente cuáles dos características producen el menor error empírico de clasificación, usando el conjunto de prueba.

7. Con las dos características determinadas en el punto anterior, visualice las superficies de probabilidad para cada clase c: $p(y = c|\underline{\boldsymbol{x}})$, de forma similar a lo ilustrado en la figura 2.

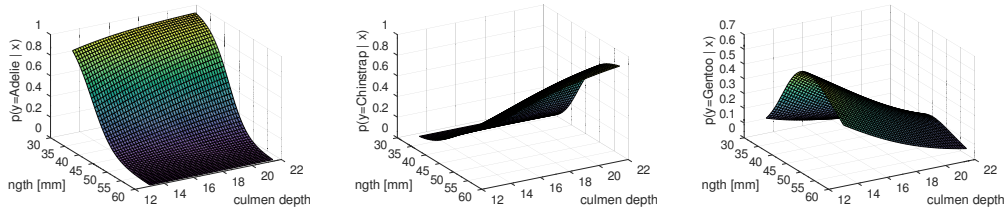

Figura 2: Superficies de probabilidad para las tres especies.

8. Visualice en el espacio de entrada las regiones en las que cada clase gana (es decir, la clase más probable), de forma similar a lo ilustrado en la figura 3.

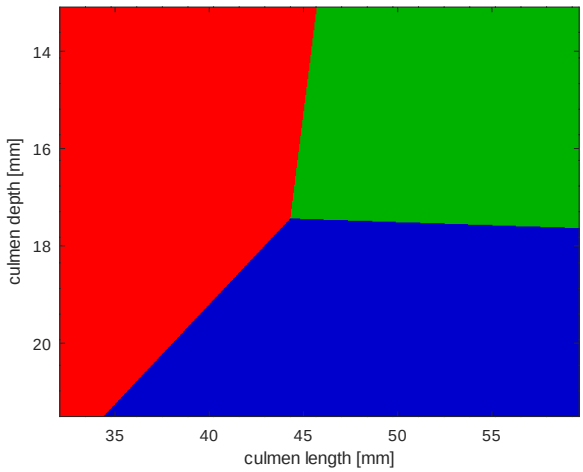

Figura 3: Regiones de las clases ganadoras para el espacio de entrada bidimensional.

Las funciones `ind2rgb` e `image` le pueden ser de utilidad, así como el código del ejemplo brindado en la lección 7 para softmax.

9. La visualización de clase ganadora tiene el inconveniente de que no queda claro qué tan seguro está el clasificador de la asignación. Si se hace la mezcla de los colores, ponderada por la probabilidad, sí es posible apreciar qué tan seguro está el clasificador.

   Agrege ahora código para producir la imagen con cada color ponderado por la probabilidad de la clase correspondiente, como se ilustra en la figura 4.

10. Puntos extra. Agregue un archivo `softmax_main2.m` con la funcionalidad de los puntos 7, 8 y 9, pero modificados de tal forma que que permita realizar fronteras de división no lineales. Puede utilizar los desarrollos hechos por usted en la tarea 2. (4 puntos)

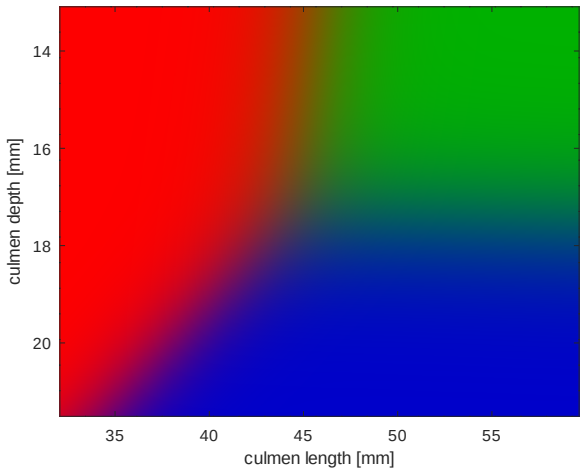

Figura 4: Ponderación de colores asignados a las clases, de acuerdo a la probabilidad de pertenecer a esa clase.

Entregable: archivos de GNU/Octave, y README con instrucciones de cómo ejecutar el código. Debe mostrarse cada punto solicitudo en figuras aparte.

Esta tarea se realiza en parejas o individual. Informar al profesor si el grupo de trabajo en esta tarea cambió respecto al de la tarea 3.

Solo lo que sea subido al tecDigital será calificado. Entregas por correo serán ignoradas, por lo que asegúrese de haber enviado correcciones a su grupo (aún si es individual) a tiempo.

Esta tarea requiere tiempo. Evite dejarla para los últimos días.

Se revisará el trabajo individual por medio de la actividad en Git.

Toda tarea debe ser resultado del trabajo intelectual propio de la persona o personas que la entregan. Además de la literatura de referencia, solo puede utilizarse el material expresamente así indicado en la tarea, lo que incluye código brindado por el profesor o indicado en los enunciados a ser utilizado como base de la tarea. Expresamente quedan excluidos como material de referencia los trabajos entregados por otros estudiantes en el mismo semestre o semestres anteriores.

Nótese que esto no elimina la posibilidad de discutir estrategias de solución o ideas entre personas y grupos, lo cual es incluso recomendado, pero la generación concreta de cada solución, derivación o programa debe hacerse para cada entrega de forma independiente.

Para toda referencia de código o bibliografía externa deben respetarse los derechos de autor, indicando expresamente de dónde se tomó código, derivaciones, etc. Obsérvese que el código entregado por el profesor usualmente ya incluye encabezados con la autoría correspondiente. Si un estudiante agrega código a un archivo, debe agregar su nombre a los encabezados si la modificación es de más del 50 % del archivo, o indicar expresamente en el código, con comentarios claros, la autoría de las nuevas líneas de código, pues a la autora o al autor del archivo original no se le debe atribuir código que no es suyo.

Si se detecta código o deducciones teóricas iguales o muy cercanas a trabajos de otros estudiantes del mismo semestre o de semestres anteriores, se aplicará lo establecido por la reglamentación vigente, en particular el Artículo 75 del Reglamento de Régimen de Enseñanza y Aprendizaje.

Modificaciones de comentarios, cadenas alfanuméricas, nombres de variables, orden de estructuras independientes, y otras modificaciones menores de código se siguen considerando como clones de código, y las herramientas automatizadas de detección reportarán la similitud correspondiente.

Los estudiantes que provean a otros estudiantes del mismo o futuros semestres soluciones de sus tareas, también son sujetos a las sanciones especificadas en la reglamentación institucional. Por lo tanto, se advierte no poner a disposición soluciones de las tareas a otros estudiantes.