Regresión Logística

Práctica de Laboratorio 2 - 2020I

Tópicos de Inteligencia Artificial

I. Pautas

- Fecha de Entrega: 16 , a través de Moodle.
- Lenguaje de Programación: Python.

II. Implementación (6.5)

Implemente las siguientes funciones:

1. `Leer_Datos`: Recibe el nombre el archivo en formato csv y retorna los datos en un `nparray`.
2. `Normalizar_Datos`: Recibe los datos en `nparray` y los convierte en datos cuya media es cero y desviación estándar es 1.
3. (0.5) `Sigmoidal`: Recibe X y θ; y devuelve el cálculo de la función Sigmoidal (también conocida como función Logística).
4. (1.5) `Calcular_Funcion_Costo`: Recibe X,y y θ; y devuelve el valor de la función de Costo de Entropía Cruzada asociado a los datos (X,y).
5. (0.5) `Calcular_Gradiente`: Recibe X, y, θ; y devuelve el gradiente asociado a los datos (X,y).
6. (0.5) `Gradiente_Descendiente`: Recibe X, y, θ, el número de iteraciones y la tasa de aprendizaje, y actualiza θ usando el gradiente descendiente. Usa las funciones: `Sigmoidal`, `Calcular_Funcion_Costo` y `Calcular_Gradiente`. Devuelve los valores actualizados de θ y el costo de cada iteración.
7. (2) `Calcular_Accuracy`: Recibe X, y y θ; y devuelve el *accuracy* (porcentaje de ejemplos correctamente clasificados) asociado a los datos (X,y).
8. (1.5) `Crear_k_folds`: Recibe un conjunto de datos normalizado y un entero k (en los experimentos, debe usar $k = 3$), y divide los conjunto en k subconjuntos disjuntos del conjunto de entrada. Cada subconjunto debe mantener la misma proporción de elementos de cada clase, es decir, si el conjunto de datos de entrada tiene el 60 % y 40 % de elementos de las clases 1 y 2, respectivamente, entonces cada uno de los subconjuntos debe el 60 % y 40 % de elementos de las clases 1 y 2. La función devuelve k conjuntos, cada uno de ellos separado en sus respectivos X_k y y_k.

III. Experimentos (12)

Importante: Para que los experimentos sean calificados sus resultados deben estar dentro del informe.

1. (4) Experimento 1: Buscar los mejores parámetros de entrenamiento para los conjuntos "C2"y "C1", usando validación cruzada (k-$fold$ *cross validation*, con $k = 3$). En la validación cruzada se entrena con unos parámetros específicos y se calcula el promedio de los *accuracies* obtenido al ejecutar el algoritmo del gradiente descendiente k veces. En cada vez, uno de los $folds$ es usado como conjunto de prueba y resto, i.e. los otros dos, como conjunto de entrenamiento. Los $folds$ son conjuntos disjuntos dos a dos del conjunto de datos original.

Dichos promedios deben ser mostrados en dos tablas, una para "C2"y otra para "C1". En cada tabla, se mostrará el *Accuracy* promedio obtenido, al variar los parámetros de entrenamiento, tal como se muestra a continuación:

Parámetros	Valores
Tasa de Aprendizaje	$0{,}01; 0{,}05; 0{,}1; 0{,}2; 0{,}3; 0{,}4$
Iteraciones	$[500, 3500]$ en incrementos de 500

2. (1.5) Experimento 2: Plotear el costo histórico de cada uno de los conjuntos de entrenamiento de "C2"y "C1".

3. (5) Experimento 3 - Clasificación multiclase: Implementar el enfoque uno-vs-uno y uno-vs-todos. Luego, encontrar los mejores parámetros de entrenamiento para el conjunto de datos "C3", usando validación cruzada (con $k = 3$) y el *accuracy* como métrica de evaluación.

IV. Informe (3)

Importante: El informe solo recibirá puntaje si está terminado, si está bien redactado y si los experimentos tienen discusión.

El informe debe estar redactado en Latex (plantilla LNCS) y tener la siguiente estructura:

- Título "Práctica de Laboratorio Nro. 2 ".
- Datos del alumno conforme la plantilla.
- Introducción
- Implementación: Hacer referencia al código.
- Experimentos y Resultados: La sección debe estar dividida en tantas secciones como experimentos existan y cada experimento debe discutido.
- Conclusiones.
- Referencias: Si las hubieran.