Ricardo Aler Mur

SEGUNDA PRÁCTICA ANÁLISIS DE DATOS R

Puntuación: 3.5 puntos

ANÁLISIS DE DATOS EN R, APLICADO AL RECONOCIMIENTO DE DÍGITOS (SEMEION)

1. INTRODUCCIÓN

Se va a trabajar con el dominio de reconocimiento de caracteres SEMEION (hermano pequeño del dominio MNIST). Se trata de aprender a clasificar dígitos (de cero a nueve) escritos a mano y representados con una matriz de 16x16 bits. El fichero original tenía 1593 dígitos escritos por 80 personas. Los niveles de grises fueron convertidos a blanco/negro. Cada persona escribió todos los dígitos de 0 a 9 dos veces, la primera vez despacio y la segunda vez, deprisa. Los datos contienen 16*16=256 atributos booleanos (256 píxeles a blanco o a negro) mas la clase (0 a 9).

Figure 1: written in normal way

Figure 2: written in fast way

Se suministra el dataframe “SEMEION_disponibles” que contiene 2/3 de los datos totales (el otro 1/3 lo he reservado para hacer un test independiente) y un código fuente que hace un análisis de ranking de atributos y aplicación de árboles de decisión, KNN y LVQ a los datos, así como transformación mediante PCA.

2. **SE PIDE:**

1. (**0.5 puntos**) Comentar el código suministrado, explicando lo que hace cada trozo de código.
2. (**0.5 puntos**) En el código suministrado no se utiliza validación cruzada sino que se evalúan los modelos con un único conjunto de validación. De la clase de evaluación de modelos sabemos que cuando se utiliza un conjunto de test (o de validación) se puede establecer un intérvalo de confianza para la binomial resultante. El código está en aula global. Modificar el código para que en lugar de imprimir el error, se imprima el intérvalo de confianza correspondiente.
3. (**1 punto**) En el código suministrado sólo se prueba un parámetro para cada tipo de modelo (cp para árboles de decisión, k para KNN y LVQ). Probar otros valores para los parámetros para ver si se consigue reducir el error. Probar también a cambiar el número de atributos seleccionados o el número de componentes que se utilizan de PCA. Aunque no se consiga reducir el error, intentar encontrar el número mínimo de atributos o de componentes PCA que son necesarios. Todo esto se puede probar a mano, o bien automatizarlo mediante bucles.
4. (**0.5 puntos**) Cuando se explicó KNN, se habló de técnicas para reducir el número de datos (edición y condensación). Existe código en Aula Global para hacer esto. Usadlo. ¿Cuánto se reducen los datos?. ¿Disminuye o se incrementa el error?
5. (**1 punto**) Para cada dato, probad a añadir nuevos atributos de la siguiente manera. Cada dato es realmente una matriz de 16x16 píxeles. Añadid 8 atributos que representen la media de píxeles a uno en cada una de las filas y 8 atributos que representen la media de píxeles a uno en cada una de las columnas. Se pueden utilizar técnicas de selección de atributos si es necesario. ¿Se consigue mejorar el error?

Es necesario entregar el código R utilizado para cada uno de los apartados (1, 2, 3, 4, 5) y una memoria donde describáis los resultados. Además, habrá que entregar un programa adicional que tenga una subrutina:

predecir <- **function** (inTest) { … return(outTest)}

que genere los mejores resultados que hayáis encontrado a lo largo de la práctica. Yo lo probaré con mis datos de test y anunciaré los resultados cuando haya corregido las memorias.

Nota adicional: si hacéis experimentación por vuestra cuenta con redes de neuronas, máquinas de vectores de soporte, random forests, autoencoders o alguna otra cosa que me propongáis se valorará con 0.5 puntos adicionales.