Proyecto final de Mineria de Datos: Clasificador para predecir enfermedades del corazón (Predict Heart Disease)

Ruiz Gómez, Víctor
Garcia Marreros, Ricardo
Esteban manco

24 de junio de 2016

1. Introducción

Nuestro objetivo es construir un clasificador que sea capaz de predecir si un paciente puede tener una enfermedad del corazón o no. Nuestro algoritmo podrá predecir que el paciente:
- No tiene enfermedad
- Tiene una angina típica o común
- Tiene una angina atípica
- Tiene una angina asintomática

Para ello, disponemos del dataset "cleveland", con datos de pacientes anteriores de los que se conoce su diagnóstico.
En total, se posee información de un total de 297 pacientes y una serie de valores por cada uno de estos (edad, sexo, frecuencia cardiaca máxima alcanzada, ...)
En total hay 64 variables con las que podemos trabajar pero en la práctica, solo usaremos 14 de estas.

2. El clasificador

El algoritmo que usaremos para clasificar los nuevos pacientes, podrá estar basado en el algoritmo K-Nearest Neighbour (KNN), o un sistema basado en reglas de chi, usando una configuración específica.

Por un lado ejecutaremos un algorítmo genético evolutivo para obtener el mejor clasificador basado en KNN, que clasifique con más acierto el conjunto de entrenamiento (el dataset cleveland proporcionado), es decir, que maximice el acuraccy rate.

Por otro, otro algoritmo genético, pero en este caso, servirá para obtener el mejor clasificador para un sistema basado en reglas de chi, sobre el dataset cleveland.

3. Algoritmo genético evolutivo y KNN

El objetivo es construir un algoritmo genético cuyos cromosomas codifiquen distintas configuraciones del KNN.
La función de ajuste que usaremos para asociar valores de fitness a los cromosomas, será proporcional al accuracy rate obtenido al aplicar KNN con la configuración codificada por cada cromosoma.

3.1. Configuración del clasificador KNN

Los parámetros que definen la configuración de nuestro clasificador KNN son los siguientes:

- El parámetro k, que podrá tomar valores en rango $[1, 32]$
- "tipoKNN", que determina si el KNN esta basado en conjuntos difusos o no (fuzzyKNN)
- El parámetro KF, que solo tendrá relevancia si el KNN es difuso y que estará siempre en el rango $[1, 32]$
- Por último, "tipoDist", que indicará la forma en la que se calcularán las distancias en el algoritmo KNN (distancia euclidea / manhattan)

3.2. Representación de los cromosomas

La representación de los cromosomas será binaria; Una secuencia de bits de cierta longitud:
- 5 bits para codificar k
- 5 bits para codificar KF
- 1 bit para codificar el tipo de KNN ("tipoKNN")
- 1 bit para codificar el tipo de distancia ("tipoDist")

En total necesitamos 12 bits para la codificación de los cromosomas.
El algoritmo genético buscará una configuración entre las 2^{12} posibles.

3.3. Función de ajuste de los cromosomas

Antes de comenzar la ejecución del algoritmo genético, se divide en primer lugar, el conjunto de ejemplos de entrenamiento en $k = 4$ particiones.
El valor de fitness asociado a un cromosoma, será obtenido aplicando el procedimiento de validación cruzada con las k particiones, usando la configuración del KNN codificada por el mismo.

El ajuste del cromosoma será la media de los "accuracy" obtenidos clasificando cada partición, tomando como conjunto de entrenamiento el resto de particiones

3.4. Configuraciones adicionales del genético

La población estará compuesta por $M = 12$ individuos. Se usa el método del torneo con $k = 3$ para la selección de progenitores.
Para cruzar pares de cromosomas, se usa la técnica 2-crossover (cruzamiento por 2 puntos).
La mutación consistirá en modificar de forma aleatoria, alguno de los genes de la población. Se mutarán en media, el 5 % de los genes de la población en cada iteración del genético.
Para seleccionar los sucesores, se toma un 30 % y un 70 % de mejores individuos adaptados (con mayor evaluación de fitness), entre los progenitores y los descendientes respectivamente.

El tamaño de la población se mantiene constante en todas las generaciones.
Por último, como criterio de finalización, se establece un número máximo de iteraciones (40)

4. Algoritmo genético evolutivo y SGBR

Ahora vamos a trabajar con otro algoritmo genético usando SGBR.
En este caso, partiremos de una base de reglas ya aprendida (tomando como datos para el aprendizaje, el conjunto de entrenamiento al completo fuzzyficado).
Posteriormente, se lanza el genético, donde cada cromosoma codificará el conjunto de reglas que se usarán para la clasificación de los ejemplos de train (selección de reglas)
Como resultado obtendremos un subconjunto de reglas, para las cuales se obtiene un mayor accuracy usándolas para clasificar los ejemplo de train.

4.1. Configuración del clasificador SGBR de Chi

Para obtener en primera instancia la base de reglas, ejecutamos el algoritmo de Chi con la siguiente configuración:
- Nº etiquetas = 3
- Producto como T-normas para el calculo de grados de compatibilidad
- Pesos de las reglas basado en la confianza penalizada con la suma de las confianzas del resto de clases.

4.2. Representación de los cromosomas

La codificación de los cromosomas será binaria. Será una secuencia de bits, tantos como reglas.
Cada bit indica si cierta regla debe seleccionarse o no.

4.3. Función de ajuste de los cromosomas

Para evaluar el fitness de los individuos, se seleccionan las reglas en base a los bits que codifican sus cromosomas.
Con ese conjunto de reglas seleccionado, se lleva a cabo la inferencia de cada ejemplo del conjunto de entrenamiento.
El valor de fitness, será el "accuracy rate" obtenido en la clasificación.

4.4. Configuraciones adicionales del genético

Se usa la misma configuración que para el algoritmo genético con KNN.

5. Resultados de las ejecución de los algoritmos genéticos

Tras varias ejecuciones del primer algoritmo genético, el clasificador KNN que obtiene mejor "accuracy rate" en test usa la siguiente configuración:
- KNN difuso
- k = 16
- KF = 21
- Distancia manhattan

El valor de fitness del cromosoma que codifica esta configuración para el clasificado KNN, es 0.483.
El "accuracy rate" obtenido sobre el conjunto de test es 0.583

Una de los soluciones obtenidas usando el 2ndo algoritmo genético, nos da un "accuracy rate" en train (valor de fitness) de 0.801 sobre el conjunto de entrenamiento, seleccionando 168 reglas de las 237 totales.

Evaluando el clasificador seleccionando esas 168 reglas sobre el conjunto de test, obtenemos un accuracy rate de "0.4833"