ugr Universidad de Granada

Clasificación con Conjuntos de Datos No Balanceados
Laboratorio de Programación en R

Minería de Datos, Aspectos Avanzados

Dpto. Ciencias de la Computación e Inteligencia Artificial
E.T.S. de Ingenierías Informática y de Telecomunicación
Universidad de Granada

Índice

1. Objetivos

Para esta sesión de laboratorio, se necesitará descargar desde PRADO2 el fichero imbalanced.R, además de los conjuntos de datos subclus.txt y circle.txt.

Para la evaluación, tendréis que subir los ficheros R imbalanced.R (completo) a la actividad correspondiente de PRADO2, antes del 6 de Febrero a las 23.59. Hay que asegurarse de que el código entregado esté suficientemente comentado. También, proporcione un análisis breve de los resultados obtenidos usando comentarios en los ficheros R.

2. Clasificación no balanceada

En este ejercicio, se implementará las estrategias a nivel de datos ROS, RUS y SMOTE para tratar con distribuciones de datos no balanceadas.

2.1 Preparación de datos y rendimiento básico de clasificación

Abra el fichero imbalanced.R. El código carga el conjunto de datos bidimensional subclus y lo visualiza. Calcula el ratio de imbalanceo (IR) y configura las particiones para un 5-folds cross validation (5-FCV), respetando el desbalanceo de clase en cada partición.

Finalmente, evalúe el rendimiento del clasificador base 3NN sobre este conjunto de datos por medio de la medida g-mean.

Hágalo también sobre el conjunto de datos circle.

2.2 Random Oversampling (ROS)

Estudie la implementación de la técnica de random oversampling (ROS) en el fichero imbalanced.R.

La aplicación de ROS debería obtener un conjunto de entrenamiento perfectamente equilibrado, duplicando aleatoriamente las instancias de la clase minoritaria.

Evalúe su rendimiento sobre el conjunto de datos subclus, por medio de un 5-FCV, utilizando el clasificador 3NN.

Hágalo también sobre el conjunto de datos circle.

2.3 Random Undersampling (RUS)

Estudie la implementación de la técnica de random undersampling (RUS) en el fichero imbalanced.R.

La aplicación de RUS debería obtener un conjunto de entrenamiento perfectamente equilibrado, borrando instancias de la clase mayoritaria de forma aleatoria.

Evalúe su rendimiento sobre el conjunto de datos subclus, por medio de un 5-FCV, utilizando el clasificador 3NN. Se visualizan las acciones de este técnica de preprocesamiento.

Hágalo también sobre el conjunto de datos circle.

2.4 Synthetic Minority Oversampling Technique (SMOTE)

2.4.1 Distancia

El fichero imbalanced.R contiene una función de distancia que calcula la distancia entre dos instancias en un conjunto de datos, teniendo en cuenta que algunas características son nominales (la distancia por característica para atributos nominales debería ser de 0 si las instancias tienen el mismo valor y de 1 en caso contrario).

2.4.2 getNeighbors

Escriba una función que encuentre los 5 vecinos de la clase minoritaria más cercano a un elemento, dados los índices de los elementos minoritarios y un conjunto de entrenamiento:

```
getNeighbors <- function(x, minority.instances, train){
        ...hacer algo  ...
        return( ...)
}
```

2.4.3 syntheticInstance

Implemente una función syntheticInstance que, dados los vecinos seleccionados de x, se escoja uno al azar y construya una instancia sintética en una posición aleatoria sobre el segmento de la línea entre x y su vecino. Para atributos nominales, la nueva instancia escoge aleatoriamente entre el valor de x o de su vecino. La función debería devolver la nueva instancia.

Consejo: considerar usar la función runif.

2.4.4 Evaluación

Poner en conjunto las funciones anteriores y usarla en la implementación del método SMOTE.

Verificar su rendimiento sobre el conjunto de datos subclus y circle.

Proporcionar visualización.

3. Borderline-SMOTE *(parte optativa)*

Una de las extensiones más conocidas de SMOTE es el algoritmo Borderline-SMOTE (Han et al., ICIC 2005). Puedes encontrar la descripción del método aquí: http://sci2s.ugr.es/keel/keel-dataset/pdfs/2005-Han-LNCS.pdf.

Extender la implementación previa de SMOTE para que ahora se realice el proceso descrito en Borderline-SMOTE1. Para ello hay que tener en cuenta varios detalles:

- La función getNeighbors debe considerar ahora ejemplos minoritarios y mayoritarios.
- Considere también calcular los 5 vecinos más cercanos.
- Los puntos minoritarios ruidosos (aquellos cuyos vecinos cercanos son TODOS de la clase mayoritaria) no se consideran.
- Calcular el conjunto DANGER de ejemplos borderline descrito en el paper descriptivo. Serán aquellos en los que más de la mitad de sus vecinos más cercanos son de la clase mayoritaria.
- Continuar el proceso restante de SMOTE solo generando instancias artificiales de los puntos contenidos en DANGER hasta equilibrar el conjunto de datos.

Verificar su rendimiento sobre el conjunto de datos subclus y circle. Proporcionar visualización.

3. Paquete 'unbalanced' y combinación de técnicas

Existe un paquete en CRAN llamado 'unbalanced' que implementa algunas de las técnicas más conocidas de preprocesamiento de datos para clasificación no balanceada. La documentación se puede encontrar en https://cran.r-project.org/web/packages/unbalanced/unbalanced.pdf.

Utilizando el paquete 'unbalanced', se pide combinar dos técnicas de undersampling con SMOTE. La primera de ellas será la combinación SMOTE + TomekLinks y la segunda será SMOTE + ENN.

Verificar su rendimiento sobre el conjunto de datos subclus y circle. Proporcionar visualización.