Big Data II: Práctica de Spark

Máster en ciencia de datos e ingeniería de computadores - UGR

25-5-2018

Mª Cristina Heredia Gómez

crstnheredia@correo.ugr.es

Índice

Dataset

El conjunto de datos utilizado para los experimentos es ECBDL_14, un subdataset de ECBDL que contiene 1 millón de instancias divididas en 499,449 para entrenamiento y 500,353 para test. El dataset tiene un total de 631 características y tiene dos clases, donde la mayoritaria representa el 70 % de los datos y la minoritaria representa el 30 %.

Ejecuciones y resultados

En los experimentos se han usado tres algoritmos base: árboles de decisión, random forest y PCARD. Para las comparativas, se ejecutaron estos algoritmos con y sin preprocesamiento previo. Los algoritmos de preprocesamiento empleado son tres: ROS, RUS, y HTE-BD del framework de Spark *NoiseFramework*. Además, para ROS se generaron varios datasets empleando diferentes ratios de oversampling, usando el siguiente script:

Listing 1: Script para generación de dataset ROS

```
#!/bin/bash
#Cluster (.jar must be in cluster)
/opt/spark-1.6.2/bin/spark-submit --master spark://hadoop-master:7077 \
--class org.apache.spark.mllib.sampling.runROS \
--total-executor-cores 8 \
--executor-memory 8g \
./target/Imb-sampling-1.1.jar \
hdfs://hadoop-master/user/spark/datasets/ECBDL14_mbd/ecbdl14.header \
hdfs://hadoop-master/user/spark/datasets/ECBDL14_mbd/ecbdl14tra.data 100 250 0 1 40 \
hdfs://hadoop-master/user/CD_76668203/spark/ECBDL14trainROSover2.data
```

Cambiando el oversampling ratio, en este caso, 40, por otros (50,70,80 y 90). Para generar el dataset con undersampling, se utilizó el siguiente script:

Listing 2: Script para generación de dataset RUS

```
#!/bin/bash
#Cluster (.jar must be in cluster)
/opt/spark-1.6.2/bin/spark-submit --master spark://hadoop-master:7077 \
--class org.apache.spark.mllib.sampling.runRUS \
--total-executor-cores 8 \
--executor-memory 8g \
./target/Imb-sampling-1.1.jar \
hdfs://hadoop-master/user/spark/datasets/ECBDL14_mbd/ecbdl14.header \
hdfs://hadoop-master/user/spark/datasets/ECBDL14_mbd/ecbdl14tra.data 250 0 1 \
hdfs://hadoop-master/user/CD_76668203/spark/ECBDL14trainRUS.data
```

Resultados

En la siguiente tabla se muestran las distintas combinaciones de algoritmos probados, así como su precisión y su tasa de aciertos en la clase positiva multiplicada por tasa de aciertos en la clase negativa. Para calcular el TPRxTNR se multiplica de forma automatizada la diagonal de cada matriz de confusión de cada experimento, que tiene la forma:

TP	FP
FN	TN

Tabla de resultados obtenida:

Algoritmo	**Precisión**	**TPRxTNR**
DT	0.7300792950687773	0.4179805
RF	0.6970065626847876	0.1147117
Pcard	0.7406576256070726	0.3779019
RUS + DT	0.7053892534824882	0.4953368
RUS + RF	0.6994639343149894	0.4943104
RUS + Pcard	0.7092982677226417	0.5053667
ROS(40) + DT	0.7192368091974525	0.2637348
ROS(40) + RF	0.6935458854777576	0.1026306
ROS(40) + Pcard	0.7380572708149555	0.335642
ROS(50) + DT	0.7329292645333902	0.4545935
ROS(50) + RF	0.6935458854777576	0.2003314
ROS(50) + Pcard	0.7329292645333902	0.406725
ROS(70) + DT	0.7291365755775066	0.4745118
ROS(70) + RF	0.7419711319419536	0.3812832
ROS(70) + Pcard	0.7362057600403261	0.4780763
ROS(80) + DT	0.7281259996704117	0.4770603
ROS(80) + RF	0.7376695198673892	0.457312
ROS(80) + Pcard	0.7307602826704408	0.4995302
ROS(90) + DT	0.72543597747167	0.4869458
ROS(90) + RF	0.7269482061671788	0.4903578
ROS(90) + Pcard	0.718446766641786	0.5066906
HTE + DT	0.7350134258765595	0.3457263
HTE + RF	0.7059951143380606	0.1536646
HTE + Pcard	0.7384401748756774	0.3240089

Conclusiones

En general aplicando previamente undersampling empeoran los resultados base en cuanto a precisión de los tres algoritmos: árbol de decisión, pcard y random forest, por lo que posiblemente estemos eliminando muestras con información discriminatoria relevante para la clasificación. En cuanto a ROS, podemos observar en la tabla de resultados que su aplicación previa con un ratio oversampling de 70 provoca mejoras significativas en cuanto a rendimiento para Random forest con respecto a no aplicar este preprocesamiento al algoritmo, mientras que para el árbol de decisión va mejor un oversampling rate de 50 y para Pcard con ninguno de los oversampling rates probados se consigue un incremento en precisión de clasificación.
En general, el algoritmo de preprocesamiento que consigue elevar tanto la precisión del árbol de decisión como de random forest es HTE, mientras que prácticamente iguala a la precisión base obtenida para Pcard, sin embargo, este algoritmo obtiene una tasa TPRxTNR inferior a la obtenida con RUS (por ejemplo) para los tres algoritmos. Esto se debe a que la tasa de instancias de la clase negativa clasificadas correctamente es inferior cuando se aplicó HTE que cuando se aplicó RUS, por lo que se penaliza mucho a HTE a pesar de que, en media, ha clasificado más muestras correctamente que RUS. Por tanto, quizás sería más adecuado usar otras medidas para evaluar el rendimiento de los algoritmos.