Niha Imam

03/23/2020

CS 484 HW 2 – Drug Activity Prediction

Username: iateyourcookie

Rank: 42

F1 - Score: 0.80

Libraries Utilized:

re, time, sklearn, imblearn

How to run:

Test and train data file must be in the same folder

The run time is ~10 minutes

Approach & Methodology

My initial approach was to just read in a data in coo matrix and run the classifiers on it. That approach was terrible since I got a F1-score of 0.15 from it. So I began rewriting my code. I extracted the data from train data and created a sparse matrix with 1 where feature is and 0 where feature isn’t. I also separated the labels too and for every label of 0 I replaced it with -1. Then I started looking into feature selections and reduction techniques. In class we talked about PCA, so I tried to use it. But the documentation on PCA and that post I found on stack overflow both recommended using TruncatedSVD works better with sparse matrix. I used TruncatedSVD to reduce the number of features from 100,000 to 5000 but I didn’t like the results so reduced the features even more to 1200. I had the TruncatedSVD set to 100 iterations with a seed of 22 but the 100 iterations made the run time worse, so I dropped it 55. After that I tried to use the classifiers, but my F1-score somehow got worse. So, I tried to do something about the imbalance nature of the data. after a lot of googling I came across SMOTE. What I gathered is that SMOTE is a technique for increasing number of cases in the dataset in a balanced way so it’s basically creating new minority instances. But looking up imbalanced learning API some of smote features had been deprecated so I used SVMSMOTE which does the same thing as SMOTE, but it also helps establish a boundary between classes which I thought would be a better choice. After that I used all my classifiers on the reduced features. I gave all my classifiers weights too. For my classifiers I used Decision Trees, Perceptron and Random Forest and Stochastic Gradient Descent. While researching I came across Random Forest which from my understanding is a variation of the decision tree classifier, so I just ended up using it as well. I came across Stochastic Gradient Descent during my research and ended up using it as well. In CS480 we were learning about it and our first project was about stochastic hill climbing so I thought it would be interesting to incorporate it. my only complaint is that its Stochastic, so the results are sometimes great and sometimes they are not that great. Which is why I used it twice once with no parameters and the other was controlled but I wasn’t as satisfied with it.

|  |  |
| --- | --- |
| Classifier | F1-score |
| Decision Tree | 0.80 |
| Perceptron | 0.52 |
| Random Forest | 0.67 |
| Stochastic Gradient Descent | 0.62 |
| Stochastic Gradient Descent Controlled | 0.58 |