HW2: Drug Activity Prediction

Bhavika Tekwani
btekwani@gmu.edu

Introduction

The specific challenges of the Drug Activity Prediction problem are feature selection due to high dimensionality and the inherent imbalance in the data. I'm using scikit-learn's `VarianceThreshold` for feature reduction, `GridSearchCV` for cross validation and parameter tuning and `SGDClassifier` for classifying drugs as Active(1) or Inactive(0).

Rank and F1-Score

As of 10/2/2016 5:36 AM, my rank was 4 and public score was 0.73.

Team Name

bolt79

Approach

Feature reduction

The train and test data contains features in the range 1 to 100000. By observation, it is evident that each drug may contain a different number of features. For the sake of simplicity, I've used `CountVectorizer` to create a binary representation of the features in each drug. 1 indicates the presence and 0 indicates the absence of a feature in a drug. Now, we have 100000 features and 800 training samples. To reduce the number of features I used the simplest technique scikit-learn has to offer - `VarianceThreshold`. Features which have a more or less consistent presence or absence in many drugs are less likely to be carrying meaningful information in the dataset i.e the provide low variance. Using cross validation, I settled on a variance threshold of 0.04. This variance threshold led to the optimum number of features being 2115. Any aggressive thresholding above 0.04 leads to a sharp decline in classifier performance. The 0 and 1 values in `CountVectorizer` have not been normalized since they are dummy indicators of the presence of a feature. Another approach I considered and tried for feature reduction and selection was Recursive Feature Elimination which proved to be computationally expensive.

Prediction

Feature reduction leads to a reduced feature space for both train and test sets which I then use for prediction. I used scikit-learn's `SGDClassifier`. Stochastic Gradient Descent (SGD) classification is efficient and scales well to an increasing number of features. I used `GridSearchCV` to find the optimal set of parameters for the reduced feature set. The penalty function used is `elasticnet` - a convex combination of L1 and L2 regularization. The loss function is Modified Huber - a smoothed form of hinge loss used in linear support vector machines. It has higher tolerance to outliers than other loss functions.

The default learning rate `eta0` is 'optimal' for classification problems. The regularization term α is usually in the range 10^{-1} to 10^{-6}. I found the optimal value of α to be 0.07 using `GridSearchCV`. The number of iterations or passes the classifier makes over the training data is set to 10000. SGD has been known to converge after observing 10^6 features. Taking this into account, the number of iterations is calculated as $10^6/n$ where n is the number of training samples. In our case, we tune the number of iterations starting from 1250, given n=800. Other classifiers I tested using `GridSearchCV` were `AdaBoostClassifier`, `RandomForestClassifier`, `SVC`, `LogisticRegression` and `DecisionTreeClassifier` but `SGDClassifier` gave me the best results.

Cross Validation

I used a combination of `StratifiedKFold` and `GridSearchCV` to narrow down on the best classifier for the problem and tune the hyperparameters for each classifier. `StratifiedKFold` was tested with 2, 5 and 10 folds considering the small size of the train data. One fold from each of these was used as the test set to compare performance. The ROC curve obtained using k=5 for StratifiedKFold is shown below.

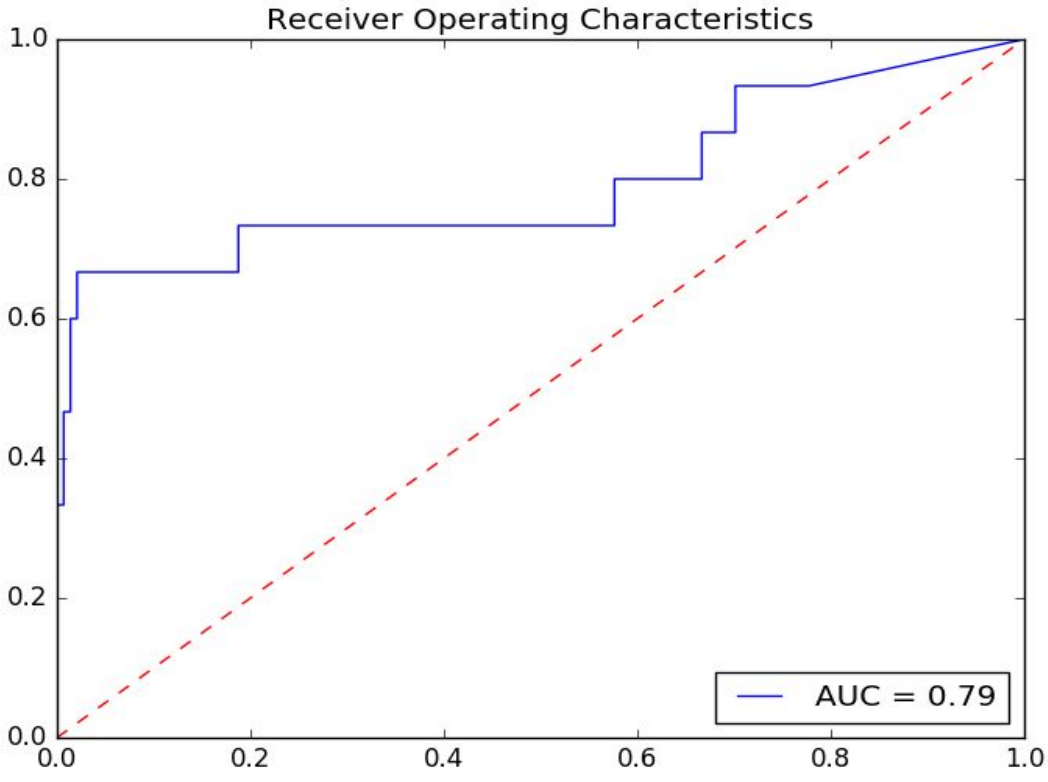

Fig. 1 ROC curve for `SGDClassifier`

Dealing with Imbalanced Data

There are several approaches for dealing with imbalanced data. Some of these approaches are oversampling, undersampling, cost sensitive classification and synthetic sample generation to balance train data. Of all these, I tried the following:

1) Synthetic Minority Over Sampling Technique (SMOTE) (oversampling)
2) RandomOverSampler (oversampling)
3) NearMiss (undersampling)
4) OneSidedSelection (undersampling)
5) TomekLinks (undersampling)

None of these methods had a positive effect on the F1-score. The final submission uses only SGD, cross validation and grid search. However, the scripts written for sampling and estimator selection are also included in the src folder.

References

Raschka, Sebastian. "Machine Learning FAQ." Sebastian Raschka. N.p., n.d. Web. 2 Oct. 2016. <http://sebastianraschka.com>.

Altini, Marco. "Dealing with Imbalanced Data: Undersampling, Oversampling and Proper Cross-validation." Marco Altini. N.p., n.d. Web. 02 Oct. 2016. <http://www.marcoaltini.com/blog/dealing-with-imbalanced-data-undersampling-oversampling-and-proper-cross-validation>.

He, Haibo, and Edwardo A. Garcia. "Learning From Imbalanced Data."IEEE Transactions on Knowledge and Data Engineering 21.9 (2009): n. pag. Web. 2 Oct. 2016. <http://www.ele.uri.edu/faculty/he/PDFfiles/ImbalancedLearning.pdf>.

"Feature Selection." 1.13. Feature Selection — Scikit-learn 0.18 Documentation. Scikit-learn.org, n.d. Web. 02 Oct. 2016. <http://scikit-learn.org/stable/modules/feature_selection.html>.

"Stochastic Gradient Descent." Stochastic Gradient Descent — Scikit-learn 0.18 Documentation. Scikit-learn.org, n.d. Web. 02 Oct. 2016. <http://scikit-learn.org/stable/modules/sgd.html>.

cdeterman. "Dealing with the Class Imbalance in Binary Classification." stackoverflow.com, 7 Oct. 2014. Web. 02 Oct. 2016. <http://stackoverflow.com/questions/26221312/dealing-with-the-class-imbalance-in-binary-classification>.

Lemaitre, Guillaume, Fernando Nogueira, and Christos K. Aridas. "Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning. N.p., 21 Sept. 2016. Web. 2 Oct. 2016. <https://arxiv.org/abs/1609.06570>.

He, Haibo, Yang Bai, Edwardo A. Garcia, and Shutao Li. “ADASYN: Adaptive synthetic sampling approach for imbalanced learning,” In IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence), pp. 1322-1328, 2008. Web.2 Oct. 2016. <http://140.123.102.14:8080/reportSys/file/paper/manto/manto_6_paper.pdf>

Chawla, N. V., K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. "SMOTE: Synthetic Minority Over-sampling Technique." Journal of Artificial Intelligence Research. Journal of Artificial Intelligence Research, 2002. Web. 2 Oct. 2016. <http://www.jair.org/papers/paper953.html>.