Tabular Data Science - Final Course Project

Eyal Cohen ID. 207947086)

Submitted as final project report for the TDS course, BIU, 2022

1 Abstract

My tool is an Outlier detection grid searcher based on PyOD(1) library algorithms for tabular data.

1.1 Problem description

There are so many Outlier detection algorithms available now, and deciding on which one to use might be a challenging task that requires lots of analysis of the data, My idea was to solve this by creating an easy Outlier detection grid searcher based on PyOD algorithms for tabular data, it provides the common data scientist a 2 lined solution for Outlier detection using grid search over the different algorithms and numbers of Outliers wanted to remove.

2 Solution

2.1 Solution overview

As said, I wanted to create a easy to use 1 or 2 liner tool for Outlier detection, after researcing I decided on using PyOD because of the number of algorithms supported there and the ease of use, at first I wanted to implement the algorithms myself, but after researching I decided that the best solution should use existing tool that proved itself time and time again, also when trying at first writing my own algorithms the run-time was significantly longer which was unacceptable for a grid searcher, a tool that is suppose to run over the data for many times.

My Grid-searcher receive on Init a training function, this function is a function of the form training_func(x,y) that return a list of scores (can be a list with one score) that would be averaged to create a final score that we want to optimize, the best score (high or low depends on configuration) will decide at the end what would be the final data (after screening the outliers), after Initializing with the training function and the different parameters(weights for the train_func output, scoring_weights, algs_to_skip etc...), all is left is to call the fit(x,y) function that will start the grid search.

The algorithms I chose are:

- Linear Model: OCSVM (One-Class Support Vector Machines), MCD (Minimum Covariance Determinant), PCA (Principal Component Analysis)
- Proximity-Based: COF (Connectivity-Based Outlier Factor), kNN (k Nearest Neighbors)
- Outlier Ensembles: LODA (Lightweight On-line Detector of Anomalies)
- Probabilistic: COPOD (Copula-Based Outlier Detection), ECOD (Unsupervised Outlier Detection Using Empirical Cumulative Distribution Functions)
- Neural Network: SO_GAAL (Single-Objective Generative Adversarial Active Learning)

Over the training the tool can print (given the verbose flag) the found Outliers and print a two dimensional PCA plot of the Outliers as red dots like seen in Fig. 1.

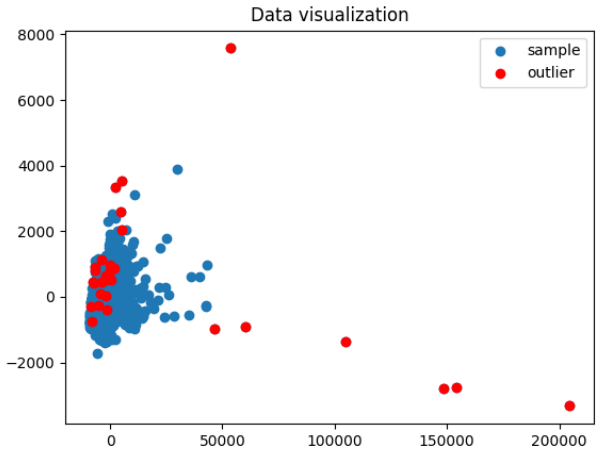

Figure 1: PCA plot of the house prices data, the red dots are the Outliers we removed.

3 Experiment results

I tested four different data-sets, one of which as suggested by Amit is a data-set that is known to contain 176 Outliers, following list is the data-sets list with the the test description, the full configuration and training funcs for each data-set can be seen in the jupyter notebook in the github directory.

Data-set	Baseline	Worst	Best	Improvement
House Prices	0.0233	0.0457 (OCSVM,20)	0.0130 (COF,30)	251%
Heart Disease	0.838	0.835 (ECOD,30)	0.845 (PCA,20)	1.1%
Heart Failure	0.848	0.831 (LODA,5)	0.856 (PCA,10)	4%
Cardiotocography	0.843	0.793 (LODA,200)	0.883 (SO_GAAL,100)	11.3%

Table 1: For House Prices data-set lower is better, the notation (i,j) near the scores is describes as (Outlier detection algorithm that was used, number of Outliers filtered), the improvement is calculated as percentages from the score of the worst score.

1. Classic House Prices - train_func uses linear regression from sklearn, it returnes as the final score a mean square error between the predicted values and the gold values.
2. Heart Disease UCI - train_func uses Logistic regression from sklearn, the final score is accuracy, recall, precision and f1 score with the following weights - 0.4, 0.0, 0.1, 0.5.
3. Heart Failure Prediction - train_func uses Logistic regression from sklearn, the final score is accuracy, recall, precision and f1 score.
4. Cardiotocography - train_func uses Logistic regression from sklearn, the final score is accuracy, recall, precision and f1 score.

To test the effectiveness of this new tool I wanted to see how much can we improve our performance when training the algorithms over the data-sets, and check the difference between the worst performing Outlier detector and the best performing one, also I wanted to test what is the difference between the best performing detector and the baseline which is not removing Outliers at all. The results can be seen on Table 1

4 Related work

As mentioned instead of implementing the Outlier filtering algorithms myself I used the PyOD library, which turned out to be a very usefull tool, Unfortunately I couldn't find any automated tools that also create a similar pipeline.

5 Conclusion

My goal in this project was to make an easy tool for Outlier detection, a task in which I think I succeeded at, like seen in Table 1 for the Cardiotocography data-set by choosing the wrong Outlier screening algorithm one can decrease its performance and that's why it's so important, also I believe that in time after

using this tool the user would get familiar with the different algorithms and will know what algorithm to use for which type of data-set to clean it's Outliers, even for me after working with this tool I can say that I know more now about how to choose the right Outlier detection algorithm for what type of data-set.

Overall this project was very interesting and new to me, I'm happy that I chose Outlier detection as the subject for this project, I can already say that I used this tool in a different project I need to submit in a different course, I think it can benefit me very much in the future.

References

[1] Y. Zhao, Z. Nasrullah, and Z. Li, "Pyod: A python toolbox for scalable outlier detection," *Journal of Machine Learning Research*, vol. 20, no. 96, pp. 1–7, 2019.