Chapter 13 - Applications of Outlier Analysis

1 Introduction

Most domains have dependency oriented data (e.g. time series, spatial data), which are harder to analyze. Supervision is often essential because unsupervised methods usually find noise, so use active learning if you can. Simple algorithms like k-nearest neighbors and the Mahalanobis method work very well. Use ensembles whenever you can.

2 Quality Control and Fault Detection Applications

If you have n widgets and know that widgets fail with probability p, you can use the probabilistic inequalities and extreme value analysis techniques from chapter 2.

Suppose we are monitoring various sensors in a running system and we want to detect faults. Extreme value analysis and abrupt change detection can be useful here. Obviously time series analysis is useful here (especially with supervision). If you treat time series as trajectories, the unusual shape detection algorithms can help. Streaming algorithms are useful here.

Suppose we have a 2D or 3D surface and we want to detect defects over time. This is spatiotemporal data. Neighborhood algorithms and unusual shape detection are useful here. You can also use temporal analysis. Obviously, supervision helps.

3 Financial Applications

Suppose we are trying to identify credit card fraud given user transactions and locations. Supervision is common here because we often have labeled data. Extreme value analysis is a good first step. Also, building per-user profiles from a short sequence of their transactions and comparing new sequences to that works well. In this case, a false negative is more dangerous than a false positive, so you should account for this. Discrete sequence methods are popular here.

Suppose we want to detect anomalous insurance claims. Note that user-specific profiles are not useful here because users do not usually make many claims. We typically extract features from the claim document and use a multidimensional outlier supervision technique.

Suppose we want to detect anomalous stock market behavior (e.g. to detect insider trading, flash crashes). News streams are useful here. Streaming methods are useful here. Time series analysis is useful.

Financial entities interact with each other. Suppose we want to detect anomalous interaction patterns. This is a temporal graph, so use those methods. Usually the edges have content, so you should extend the methods from chapter 12 to account for this.

4 Web Log Analytics

Suppose we log access to a website and want to detect anomalies. Domain knowledge helps here (e.g. lots of accesses to login page are suspicious). Position and combination outliers are useful here. Hidden Markov Models are popular here.

5 Intrusion and Security Applications

Suppose we want to detect a break-in into a computer system or network. If you have a single host being monitored, this is discrete sequence analysis (the OS system calls) and is very similar to the web log analytics case. For the network case, you could have various kinds of data (e.g. the packets), so it is more like multidimensional outlier detection because the temporal link is weaker. Some features, like number of bytes in a packet are useful. This is also a streaming situation. You also need to be able to detect novel classes.

6 Medical Applications

Given some sensor data, we want to detect if a patient has a disease. This is like fault diagnosis, but in the medical domain. Use the methods from chapter 9, if the sensors produce a time series. Supervised methods are very useful here.

We may want to detect anomalies in imaging data (e.g. MRI scan). This is spatial data where we are looking for anomalous shapes. If you have a patient history of MRIs, you can treat it as spatiotemporal data. If you know what anomalous shapes are, provide them as supervision.

7 Text and Social Media Applications

Given a set of documents (e.g. user posts), we are looking anomalous documents. See chapter 8 for techniques applicable here.

Given a stream of documents (e.g. emails), we want to identify spam. This is basically supervised text classification.

Given a social network with links, identify the noisy/spam links. See chapter 12 for linkage outliers techniques.

Give an evolving network with text in nodes, identify anomalous regions. Again, see chapter 12 for community change techniques.

8 Earth Science Applications

Suppose we have sea surface temperatures and we are looking for unsual spatial variations, strange temperature shapes, unexpected changes in temperature, and the relationship of temperature to weather patterns. This is spatial or spatiotemporal data, so neighborhood methods, unusual shape detection, and change detection are useful here.

If you have multiple behavioral attributes (e.g. temperature, wind speed), you can handle each one independently and combine the anomaly scores.

9 Miscellaneous Applications

We may want to clean a dataset by removing outliers from it. The autoregressive models from chapter 9 are good for time series and PCA is good for multidimensional data.

Given entities with trajectories, we want to find anomalies. Trajectory methods from chapter 11 work well here. Streaming algorithms may be relevant here.

Given a set of images (or stream of images), we want to find unusual shapes, changes in the temporal pattern of images, and prediction of rare classes (assuming we have some labeled data). You can treat this as spatiotemporal data. Or, if you have good image representations (i.e. image embeddings), you can treat it as a multiple time series problem.

10 Guidelines for the Practitioner

Make sure to normalize your data, or techniques like proximity methods or linear models will not work. Give your attributes zero mean and unit variance.

Many outlier methods may find noise, not interesting outliers. If you can filter out noise with domain knowledge, do it.

It's rare that your dataset will be in an immediately usable format, so you will likely want to do some feature extraction.

If you have domain knowledge, incorporate it into your algorithm. This is a way to make up for the fact that we do not have supervision. You can do this by adjusting transition probabilities for a Hidden Markov Model, tweaking your cost function (e.g. if false positive is not as bad as false negative), designing distance functions, crafting good features, providing your own sequences as comparison tokens in discrete sequence processing.

If you have labeled data, use it.

Do data visualization (e.g. see chapter 3) to get an idea of what kinds of models would work.

Add a human in the loop and do active learning if you can. If you start with an unsupervised algorithm, you can pick out some examples for the human to label and then you can switch to a supervised algorithm.

Using outlier ensembles helps overcome the limitations of a single algorithm run.

Simple algorithms on large datasets tend to beat complicated algorithms. Don't underestimate k-nearest neighbors, the Mahalanobis method, and the isolation forest. The latter two should be ensembled (notice they are almost parameter-free, too).

Your choice of parameters can impact performance, especially when you have an unstable algorithm. LOF, for example, tends to struggle when you pick bad parameters.

The linear and nonlinear Mahalanobis methods work quite well, especially when they are run on kernel matrices. Ensemble them for even better performance.

The exact and average k-nearest neighbors work quite well, and usually beat LOF. They are robust across many datasets. The key problem here is scaling to large datasets, so you'll need to use indexing/pruning or with variable subsampling (see chapter 6). Rotated bagging also helps here.

Subspace histogram methods (e.g. RSHash and isolation forest) work very well. Their only downside is they tend to prefer outliers that lie at the interior regions of the dataset. They also struggle with clusters of anomalies.

If you use an ensemble of Mahalanobis, k-nearest neighbors, and isolation forest (each of them is also ensembled), you get a powerful method called TRINITY.

11 Resources for the Practitioner

The KDD Nuggets website links to some outlier detection resources. The ELKI repo has implementations of some algorithms. Python's scikit-learn implements a bunch of algorithms too.

12 Conclusions and Summary

We've talked about various applications of outlier analysis and suggestions for how to approach different outlier analysis problems. In short, use an ensemble of k-nearest neighbors, Mahalanobis method, and subspace histograms (e.g. isolation forest).