读书笔记（二）

黄怀宇

2020 年 12 月 31 日

1 算法原理与实现

1.1 Principal Component Analysis

此算法是常见的数据降维方法，对原数据进行线性变换，并且找出信息量大的 Principal Component，去除非 Principal Component（相当于异常检测中的 anomaly/outlier）。

1.1.1 原理

首先，对原始数据进行中心化和归一化，分别使之后的公式描述更简洁，不同变量的方差变化尺度控制在相同的范围内。假设原始数据是一个 $m \times n$ 的矩阵，m 表示数据样本的数量，n 表示每一条数据样本的特征数目。将预处理之后的矩阵表示为 X，则 PCA 的主要步骤如下：

1. 计算协方差矩阵 $C = \frac{1}{m-1} X^T X$

2. 求解协方差矩阵的特征值 $\lambda_1, \lambda_2, \ldots, \lambda_i$ 和特征向量 $\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e_i}$

3. 按照特征值从大到小的顺序，将特征向量从左至右排列，将前k个特征向量组成的矩阵表示为 P_k

4. 将 X 映射到低维的 k 维空间（ $k \ll n$ ），则映射之后的数据 $Y_k = XP_k$

然后，我们要把 PCA 用于异常检测，如果把所有数据样本的重构用矩阵的形式表示出来，则得到重构公式：$X' = Y_k P_k^T$ 。对于某一个特征向量 $\mathbf{e_j}$，数据样本 x_i 在该方向上的偏离程度 d_{ij} 可以用$d_{ij} = \frac{\left(\mathbf{x_i^T} \cdot \mathbf{e_j}\right)^2}{\lambda_j}$计算：这里的 λ_j 主要起归一化的作用，这样可以使得不同方向上的偏离程度具有可比性。在计算了数据样本在所有方向上的偏离程度之后，为了给出一个综合

的异常得分，最自然的做法是将样本在所有方向上的偏离程度加起来，即：$Score\left(\mathbf{x_i}\right)=\sum_{j=1}^{n}d_{ij}=\sum_{j=1}^{n}\frac{\left(\mathbf{x_i^T}\cdot\mathbf{e_j}\right)^2}{\lambda_j}$ 。这个公式只是计算异常得分的一种方式。也有一些算法采取了略微不同的做法，比如，有的只考虑数据在前 k 个特征向量方向上的偏差，或者只考虑后 r 个特征向量方向上的偏差，即：$\sum_{j=1}^{k}d_{ij}>C_1 \quad \text{or} \quad \sum_{j=n-r+1}^{n}d_{ij}>C_2$ 。这里的 C_1 和 C_2 是人为设定的两个阈值，如果得分大于阈值则判断为异常。

一般而言，前几个特征向量往往直接对应原始数据里的某几个特征，在前几个特征向量方向上偏差比较大的数据样本，往往就是在原始数据中那几个特征上的极值点。而后几个特征向量有些不同，它们通常表示某几个原始特征的线性组合，线性组合之后的方差比较小反应了这几个特征之间的某种关系。在后几个特征方向上偏差比较大的数据样本，表示它在原始数据里对应的那几个特征上出现了与预计不太一致的情况。到底是考虑全部特征方向上的偏差，前几个特征向量上的偏差，还是后几个特征向量上的偏差，在具体使用时可以根据具体数据灵活处理。

前面提到，PCA用于异常检测时候，还有一种思路是基于重构误差的。直观上理解，PCA提取了数据的主要特征，如果一个数据样本不容易被重构出来，表示这个数据样本的特征跟整体数据样本的特征不一致，那么它显然就是一个异常的样本。对于数据样本 x_i , 假设其基于 k 维特征向量重构的样本为 $\mathbf{x'_{ik}}$, 则该数据样本的异常得分可以用如下的公式计算：

$$\begin{aligned}&\text{Score}\left(x_i\right)=\sum_{k=1}^{n}\left(|\mathbf{x_i}-\mathbf{x'_{ik}}|\right)\times ev(k)\\&ev(k)=\frac{\sum_{j=1}^{k}\lambda_j}{\sum_{j=1}^{n}\lambda_j}\end{aligned}$$

上面的公式考虑了重构使用的特征向量的个数 k 的影响，将 k 的所有可能做了一个加权求和，得出了一个综合的异常得分。 [1]

1.1.2 评价

此算法在降维的同时做了异常检测，可以用于可视化高维数据，辅助于人工检测。

1.2 One Class Support Vector Machine

1.2.1 原理

寻找一个超平面将样本中的正例圈出来。这里选择 SVDD(Support

Vector Data Description by Tax and Duin)，用一个球体而不是平面来把正样例圈出来。

设超球体的中心为 a，半径 R ¿ 0，体积 R^2 被最小化，中心 a 是支持向量的线性组合；要求所有到数据点 x_i 中心的距离严格小于 R，同时构造一个惩罚系数为 C 的松弛变量 ζ_i，优化问题如下：

$$\begin{gathered}\min_{R,a} R^2 + C\sum_{i=1}^{n}\zeta_i \\ \text{s.t. } \|x_i - a\|^2 \leq R^2 + \zeta_i, i = 1, \ldots, n \\ \zeta_i \geq 0, i = 1, \ldots, n\end{gathered}$$

在采用拉格朗日算子求解之后，可以判断新的数据点 z 是否在类内，如果z到中心的距离小于或者等于半径。采用 Gaussian Kernel 做为两个数据点的距离函数：

$$\|z - x\|^2 = \sum_{i=1}^{n} a_i \exp\left(\frac{-\|z - x_i\|^2}{\sigma^2}\right) \geq -R^2/2 + C_R$$

[2]

1.2.2 评价

核函数计算比较耗时，在海量数据的场景用的不多，适合数据量较小且只检测两类的情境下用。

1.3 统一实现

```
from pyod.utils.data import generate_data,
 get_outliers_inliers
from pyod.utils.example import visualize
import numpy as np
from scipy import stats
import matplotlib.pyplot as plt
import matplotlib.font_manager

from pyod.models.iforest import IForest
from pyod.models.abod import ABOD
from pyod.models.lof import LOF
```

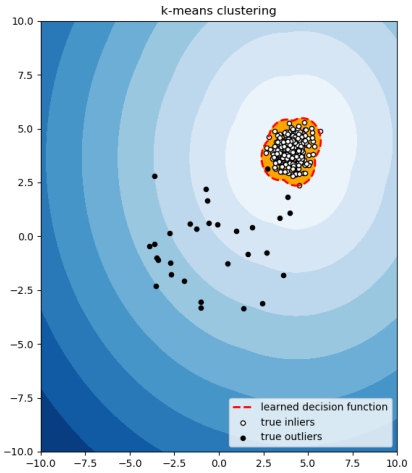
```
from pyod.models.cblof import CBLOF
from pyod.models.iforest import IForest
from pyod.models.pca import PCA
from pyod.models.ocsvm import OCSVM

data_size = 300
outlier_fraction = 0.1

X_train, Y_train = generate_data(n_train=data_size,
 train_only=True, n_features=2, behaviour="new")
x_outliers, x_inliers = get_outliers_inliers
 (X_train,Y_train)
n_inliers = len(x_inliers)
n_outliers = len(x_outliers)
F1 = X_train[:,[0]].reshape(-1,1)
F2 = X_train[:,[1]].reshape(-1,1)
xx , yy = np.meshgrid(np.linspace(-10, 10, data_size),
 np.linspace(-10, 10, data_size))
plt.scatter(F1,F2)
plt.xlabel('F1')
plt.ylabel('F2')

classifiers = {
        'k-means clustering' : CBLOF
         (contamination=outlier_fraction),
        'Isolation forest': IForest
         (contamination=outlier_fraction),
        'Local Outlier Factor' : LOF
         (contamination=outlier_fraction),
        'Principal component analysis' : PCA
         (contamination=outlier_fraction),
        'One Class Support Vector Machine' : OCSVM
         (contamination=outlier_fraction)
```

```
}

class_nums = len(classifiers)
plt.figure(figsize=(10, 10))

for i, (clf_name,clf) in enumerate
 (classifiers.items()) :clf.fit(X_train)

    # predict raw anomaly score
    scores_pred = clf.decision_function(X_train)*-1

    # prediction of a datapoint category outlier
    # or inliery_pred = clf.predict(X_train)

    n_inliers = len(y_pred) - np.count_nonzero(y_pred)
    n_outliers = np.count_nonzero(y_pred == 1)

    print('Algorithm: ', clf_name, '    OUTLIERS: ',
     n_outliers, '  INLIERS: ',n_inliers)

    # visualize
    threshold = stats.scoreatpercentile
     (scores_pred,100 *outlier_fraction)
    Z = clf.decision_function
     (np.c_[xx.ravel(), yy.ravel()]) * -1
    Z = Z.reshape(xx.shape)
    subplot = plt.subplot(1, class_nums, i + 1)
    subplot.contourf(xx, yy, Z, levels = np.linspace
     (Z.min(), threshold, 10),cmap=plt.cm.Blues_r)
    a = subplot.contour(xx, yy, Z,
     levels=[threshold],linewidths=2, colors='red')
    subplot.contourf(xx, yy, Z,
     levels=[threshold, Z.max()],colors='orange')
```

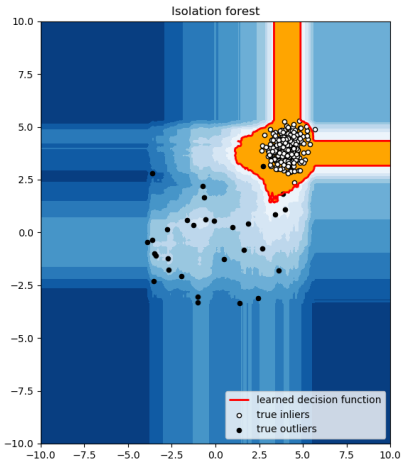
```
        b = subplot.scatter(X_train[:-n_outliers, 0],
         X_train[:-n_outliers, 1],
         c='white',s=20, edgecolor='k')
        c = subplot.scatter(X_train[-n_outliers:, 0],
         X_train[-n_outliers:, 1],
         c='black',s=20, edgecolor='k')
        subplot.axis('tight')
        subplot.legend(
            [a.collections[0], b, c],
            ['learned decision function', 'true inliers',
             'true outliers'],
            prop=matplotlib.font_manager
             .FontProperties(size=10),
            loc='lower right')
        subplot.set_title(clf_name)
        subplot.set_xlim((-10, 10))
        subplot.set_ylim((-10, 10))
plt.show()
```

```
[Running] python -u "d:\diplomaProject\code\note2\compare1.py"
Algorithm:  k-means clustering     OUTLIERS:  30   INLIERS:  270
Algorithm:  Isolation forest     OUTLIERS:  30   INLIERS:  270
Algorithm:  Local Outlier Factor     OUTLIERS:  26   INLIERS:  274
Algorithm:  Principal component analysis     OUTLIERS:  30   INLIERS:  270
Algorithm:  One Class Support Vector Machine     OUTLIERS:  30   INLIERS:  270

[Done] exited with code=0 in 198.375 seconds
```

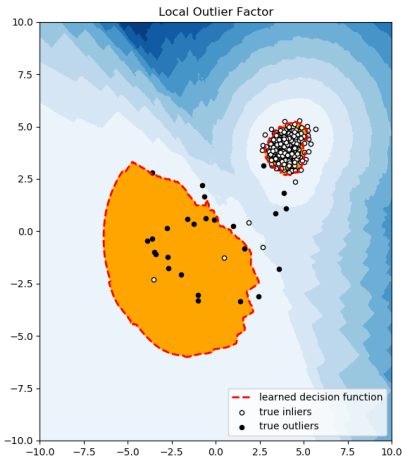
图 1: Accuracy



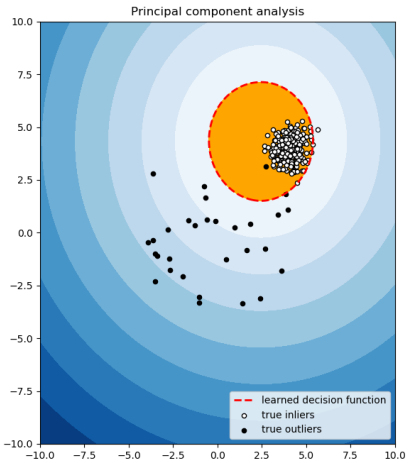
(a) Algorithm1



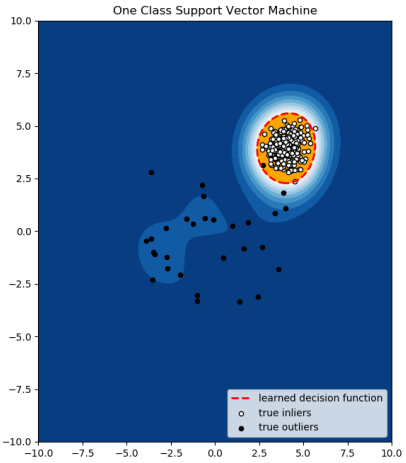
(b) Algorithm2



(c) Algorithm3



(d) Algorithm4



(e) Algorithm5

图 2: Tested Algorithm

可以发现在精度方面基准测试的结果可以和如下论文的结果一致，结论是：基于 LOF 的异常检测算法的精确度最高，速度适中，综合性能较其它算法更好。

Alg.	accuracy	deterministic	sensitivity	speed
k-NN	++	++	+	o
LOF	++	++	+	o
COF	–	++	+	o
INFLO	o	++	+	o
LoOP	++	++	+	o
LOCI	o	++	++	– –
aLOCI	–	– –	– –	o
CBLOF	– –	o	o	+
uCBLOF	++	o	o	+
LDCOF	–	o	o	+
CMGOS-Red	o	o	o	+
CMGOS-Reg	o	o	o	+
CMGOS-MCD	–	–	–	– –
HBOS	+	++	o	++
rPCA	o	++	+	+
oc-SVM	o	+	+	– –
η-oc-SVM	o	+	+	– –

图 3: Fully benchmark
[3]

2 其它算法分类与概述

2.1 选择无监督算法的必要性

监控视频异常检测主要面临以下挑战：1)异常事件定义的模糊性。正常样本与异常样本之间没有明确的划分边界。2)异常事件定义的场景依赖性。同一事件在不同的场景下具有不同的异常属性。3)异常事件的稀少性、多样性、不可穷举性。4)训练样本中包含噪声, 给样本信息带来干扰。5)由于数据的隐私性，目前可用的公开数据集较少。异常样本的稀少性、不可穷举性限制了有监督算法在该领域的应用，所以选择的算法以无监督为主。[4]

2.2 算法分类与概述

2.2.1 传统机器学习方法

1. 点模型:

聚类判别算法首先将正常样本聚类，然后将测试集中远离聚类中心的点、属于小聚类的点判断为异常。常用的聚类算法有：k-means、k-medoids、模糊c-means、Gauss混合模型(Gaussian mixed model, GMM)。常用的判断异常的方式有：测试样本到聚类中心的距离、一类支持向量机(one-class support vector machine, OC-SVM)、k最近邻(k-nearest neighbors, KNN)与核密度估计(kernel density estimation, KDE)。

共发判别算法主要基于主题模型检测异常。算法首先将正常样本划分为若干主题，然后将不属于任何正常主题的样本判断为异常。常用的主题模型算法是隐Dirichlet分配(latent Dirichlet allocation, LDA)与层次Dirichlet过程(hierarchical Dirichlet processes, HDP)。分别使用不同的特征结合LDA检测异常。HDP常与其他模型复合使用。

在传统机器学习方法中，基于重构判别的算法主要有两种：主成分分析(principal components analysis, PCA)法与SC法。它们的思路是：正常样本存在公共因子，使得正常样本可由这些公共因子重构，而异常样本则不能。从特征空间的角度看，它们都是用低维子空间来拟合正常样本在特征空间的分布。不同的是，PCA法用一个单一的低维子空间描述正常样本的分布，SC法通过一个过完备的字典与稀疏约束，用多个低维子空间的并集描述正常样本的分布。

2. 序列模型

在传统机器学习方法中，用于异常检测的序列模型主要是MDT与HMM，它们的判别都是生成概率判别。主要相关文献及算法如下：

MDT算法由K个线性动态系统(linear dynamic system, LDS)组成，用以捕捉正常样本的K种特征转移规律，当测试样本不符合其中任何一个特征转移规律时，将其判断为异常。

用无限HMM(infinite HMM, iHMM)模型结合Markov链Monte Carlo(Markov chain Monte Carlo, MCMC)采样或者变分Bayes公式来检测异常。用多观测HMM(multi-observation HMM, MOHMM)算法检测异常。将多个时空近邻的HMM进行耦合，构建耦合HMM(coupled HMM, CHMM)，使算法结合了更多的时空关联信息。将多个HMM集成检测异常。

3. 图模型

图模型算法利用时空块之间的关联关系检测异常。图推断判别算法主要利用Markov随机场(Markov random field, MRF)检测异常。提取视频时空块的混合概率PCA(mixture of probabilistic PCA, MPPCA)特征构建MRF模型，然后基于节点与节点间的推断关系检测异常。

图结构判别将图的拓扑结构作为一种新的特征检测异常。将时空近邻的特征点相连接构建图结构，然后将图的拓扑结构映射到低维流形中，在低维流形中检测异常。用视频中时空近邻的兴趣点构建图结构，以图的拓扑结构作为新的特征，通过构建图结构相似性算子实现图结构的聚类从而检测异常。

4. 复合模型

传统机器学习方法中的复合模型主要是将点模型与序列模型结合，使算法既能检测样本的分布异常，又能检测样本的转移规律异常。将LDA与HMM结合，将HDP与HMM、OC-SVM结合，将HDP与Gauss过程(Gaussian process, GP)、HMM算法结合，将SVM与HMM结合。

2.2.2 混合方法

深度特征比手工特征具有更强的描述能力。在混合方法阶段，算法使用深度特征代替手工特征，然后用传统机器学习方法检测异常。在该发展阶段，得到发展的模型主要是点模型，应用的判别主要是聚类判别、重构判别和其他判别。

2.2.3 深度学习方法

在深度学习方法阶段，算法将特征提取步骤与模型训练步骤结合到一起，用端到端的方法检测异常。

1. 点模型

深度学习方法中的聚类算法主要有：自组织映射(self-organizing map, SOM)、生长的神经气(growing neural gas, GNG)、Gauss混合全卷积VAE(Gaussian mixture fully convolutional VAE, GMFC-VAE)、一类神经网络(one-class neural network, OC-NN)。其中：SOM、GNG通过训练将不同样本映射到不同聚类实现端到端的聚类；GMFC-VAE通过变分的方式，用GMM拟合特征空间的分布；OC-NN是一种针对异常检测的神经网络，该算法将正常样本映射到一个超球体内、异常样本映射到超球体外，通过不断收缩超球体来增强网络的异常检测能力。

2. 序列模型

在深度学习方法中，常用的序列模型主要是循环神经网络(recurrent neural network, RNN)和长短期记忆(long short-term memory, LSTM)网络，常用的异常判别主要是预测误差判别。

3. 复合模型

将点模型与序列模型复合可以使算法同时捕获样本在样本空间的分布异常与转移规律异常。 [4]

2.3 算法对比分析

2.3.1 模型对比分析

不同模型的算法在异常检测中有不同的侧重点，本小节归纳了不同模型的特点，如表 1 所示。

表 1 模型对比分析

模型分类	优点	缺点
点模型	利用特征空间中的分布信息，检测特征点的分布异常	1)没有体现特征点间的时空关联关系； 2)不能检测特征点的时序异常
序列模型	利用特征点间的转移规律，检测特征点的时序异常	1)没有利用特征空间的分布信息； 2)在时空信息中仅利用了时间信息，没有利用空间关联
图模型	利用特征点间的时空关联关系，检测特征点间的时空关联异常	1)没有利用特征空间的分布信息； 2)只能对近邻时空的时空块建模，无法对长时间的特征转移规律建模

图 4: 模型对比分析

[4]

从表 1 的对比分析可以发现，不同模型对时空信息的利用程度不同，且依据异常事件的不同属性检测异常。因此，不同模型在异常检测上具有互补性，可以通过将不同模型相结合来提升算法的异常检测效果。

2.3.2 判别对比分析

聚类判别与重构判别在各个发展阶段均得到了关注，本小节分析了它们在不同发展阶段的特点, 具体见表 2。

表 2 不同发展阶段的聚类判别与重构判别的对比分析

	在不同发展阶段的相同点	在不同发展阶段的不同点	共同缺点
聚类判别	都是通过聚类描述特征空间的分布	应用特征不同；实现聚类的方法不同	特征未针对异常检测任务进行优化
重构判别	用低维子空间/流形来描述正常样本的分布，通过向子空间/流形投影，利用重构误差判断样本是否属于正常样本子空间/流	传统机器学习方法是用低维子空间来描述正常样本在特征空间中的分布，投影算子是线性函数；深度学习方法是用低维流形来描述正常样本在样本空间中的分布，投影算子是	学习到的低维子空间/流形相对于正常样本的真实分布存在冗余，落在冗余区的异常样本会被误判

图 5: 判别对比分析

[4]

参考文献

[1] 刘腾飞, “机器学习-异常检测算法（三）：principal component analysis,” 2020. https://zhuanlan.zhihu.com/p/29091645.

[2] 习翔宇, “One-class svm介绍,” 2018. https://zhuanlan.zhihu.com/p/32784067.

[3] M. Goldstein and S. Uchida, “A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data,” *PLoS ONE*, vol. 11, 2016.

[4] Z. Y. WANG Zhiguo, “Anomaly detection in surveillance videos: A survey,” *Journal of Tsinghua University (Science and Technology)*, vol. 60, 2020.