GOPS2018
Shenzhen

GOPS

全球运维大会 2018

2018.4.13-4.14 中国·广东·深圳·南山区 圣淘沙大酒店（翡翠店）

GOPS2018
Shenzhen

AIOps 亮剑网络运维-ISP流量异常检测

谭学士 360网络开发工程师

目录

POWER BY 360

360安全卫士
360杀毒
360导航
花椒直播
NEW
快视频
360安全浏览器
360搜索
360良医
手机助手
360 防骚扰大师
影视大全
360影视
360百科
360商机
360问答
360学术
360淘金
360借条
360奇云
360娱乐
你财富
贷款
360贷款导航
360信用卫士
360股票
360游戏
360商城
i360mall.com
360社区

OUR OPS

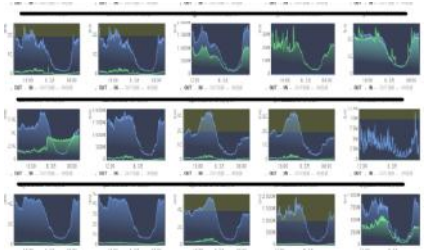
月活用户 5.15亿

大陆120 /香港1/美国1

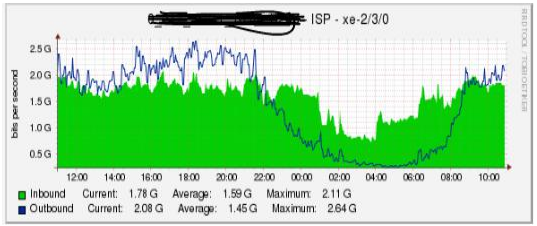
ISP 带宽 3.5T

我们对业务中断 0容忍！ 我们要洞察网络中 任何 异常！

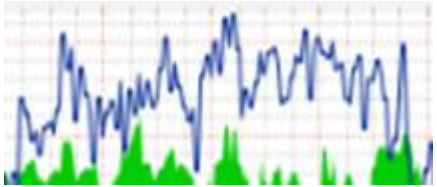
DC中ISP出口流量特征和挑战



多业务混杂，整体呈现周期性



流量波动大、频繁



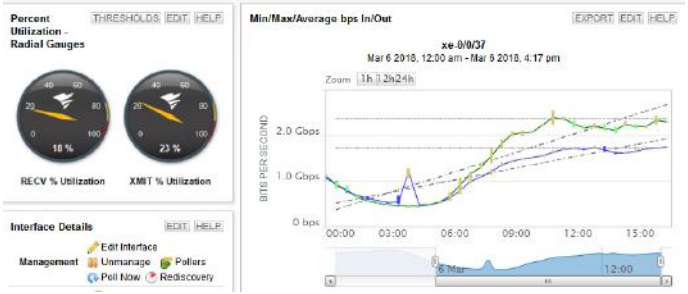
局部看没有规律

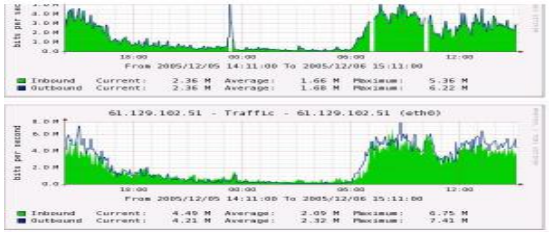
定义

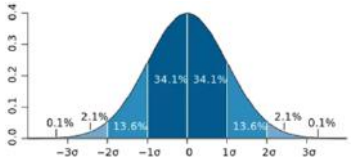
发现

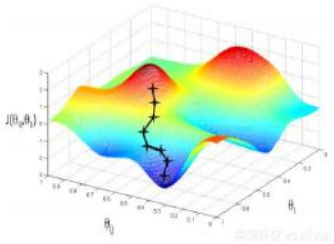
定位

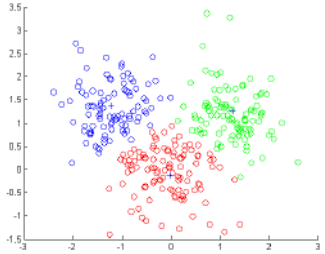
传统监控 VS 算法+机器学习

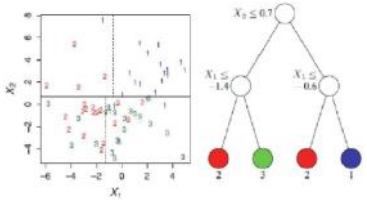


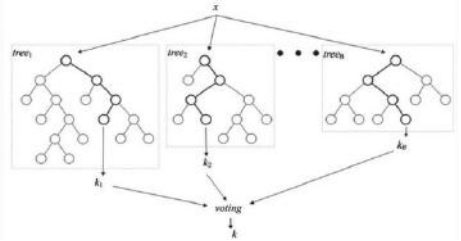






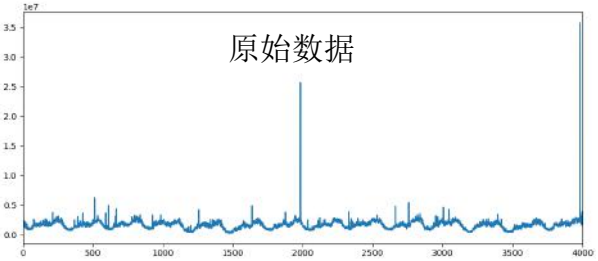


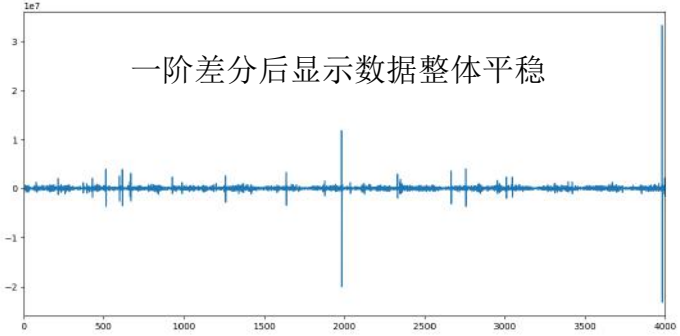


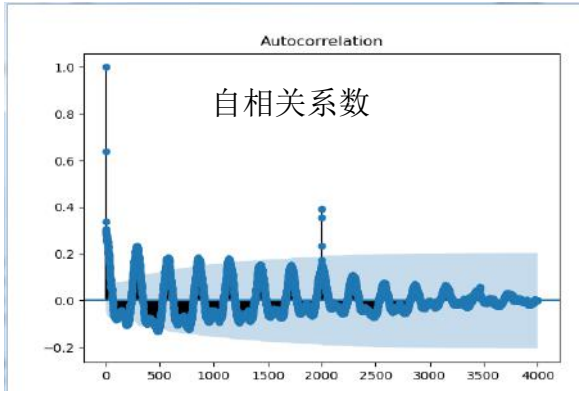


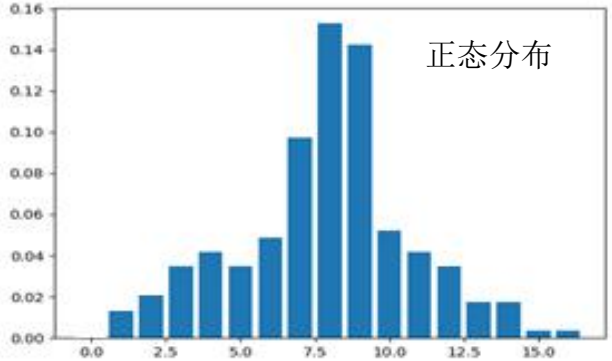
目录

平稳性检验与分布



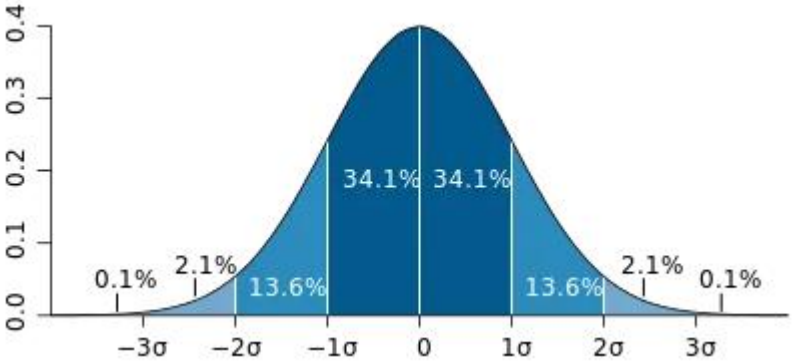






3-sigma

正态分布： $f(x) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$




```python
import numpy as np
import pandas as pd

def stddev_from_average(timeseries):

    series = pandas.Series([x[1] for x in timeseries])
    mean = series.mean()
    stdDev = series.std()
    t = tail_avg(timeseries)

    return abs(t - mean) > 3 * stdDev
```

优点：简单高效　缺点：敏感度偏高

EWMA(指数加权移动平均)

算法表示：

$$EWMA(t) = \lambda Y(t) + (1-\lambda)\, EWMA(t-1) \quad \text{for } t = 1, 2, \ldots, n.$$

设计权重系数λ， 0< λ <1， λ越大 Y(t)越大，t-1时刻就越小

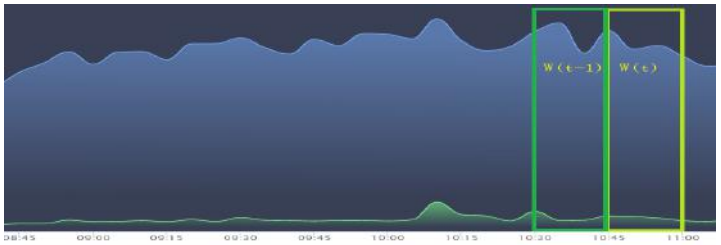
考虑趋势

环比振幅

GOPS2018
Shenzhen

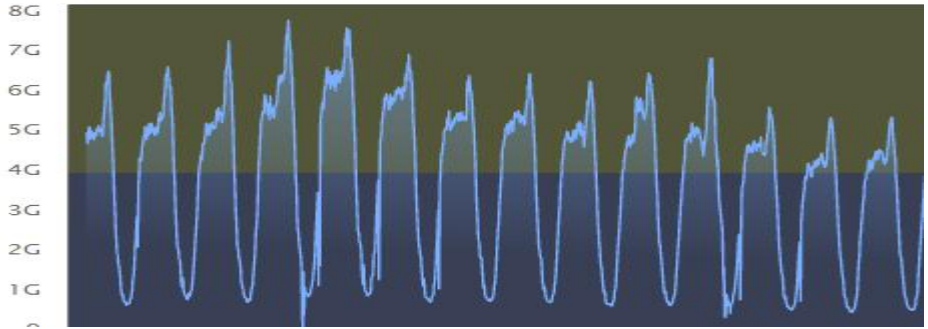
方法1： r(t) = abs($\frac{x_{(t)}}{x_{(t-1)}}$)

方法2： r(t) = abs($\frac{\mathrm{mean}(w_{(t)})}{mean(w_{(t-1)})}$)

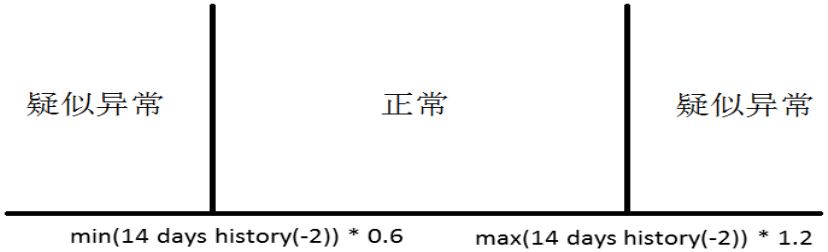


方法2 采用时间窗口，可有效吸收瞬时波动，但牺牲了敏感性

动态阈值



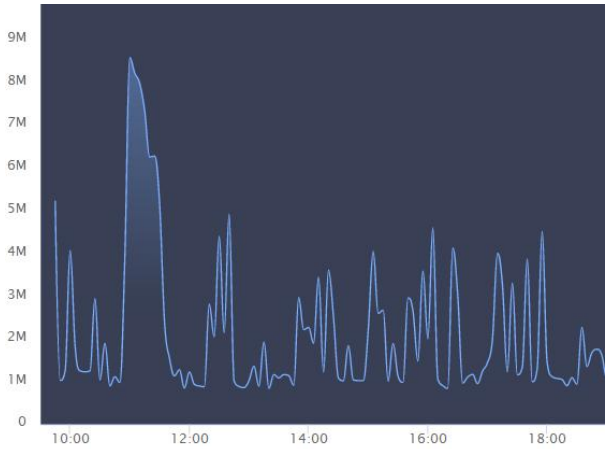
优点：
阈值动态变化
过滤单次波动对阈值的影响



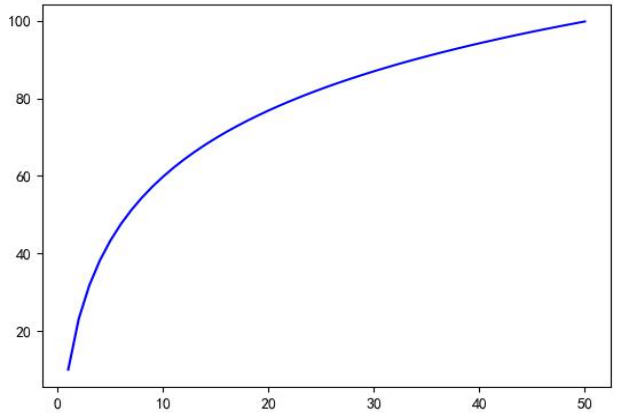
缺点：
无法发现阈值内的大幅波动异常
多次超历史的波动会影响到阈值

-2代表取值时，取倒数第2大和倒数第2小

小流量监控优化



小流量大波动

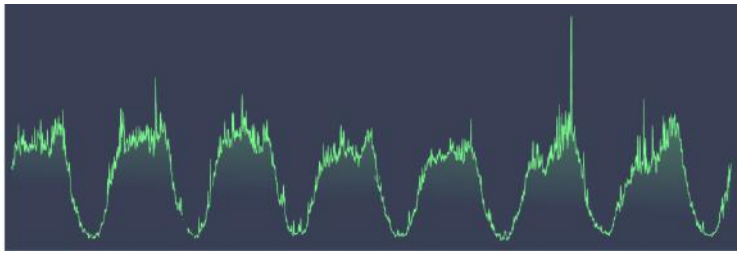


阈值曲线

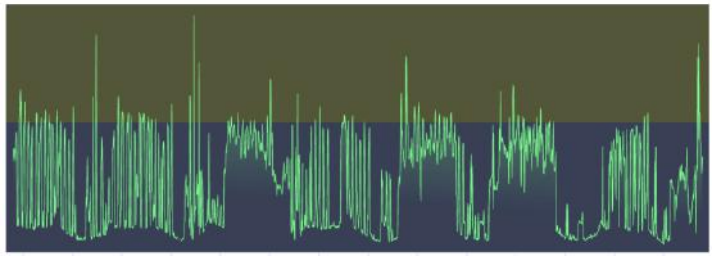
设计曲线函数：y = w * ln(x + b)

我们的参数：b = 0.4812，w = 25.4566

算法为王，为何还要机器学习？



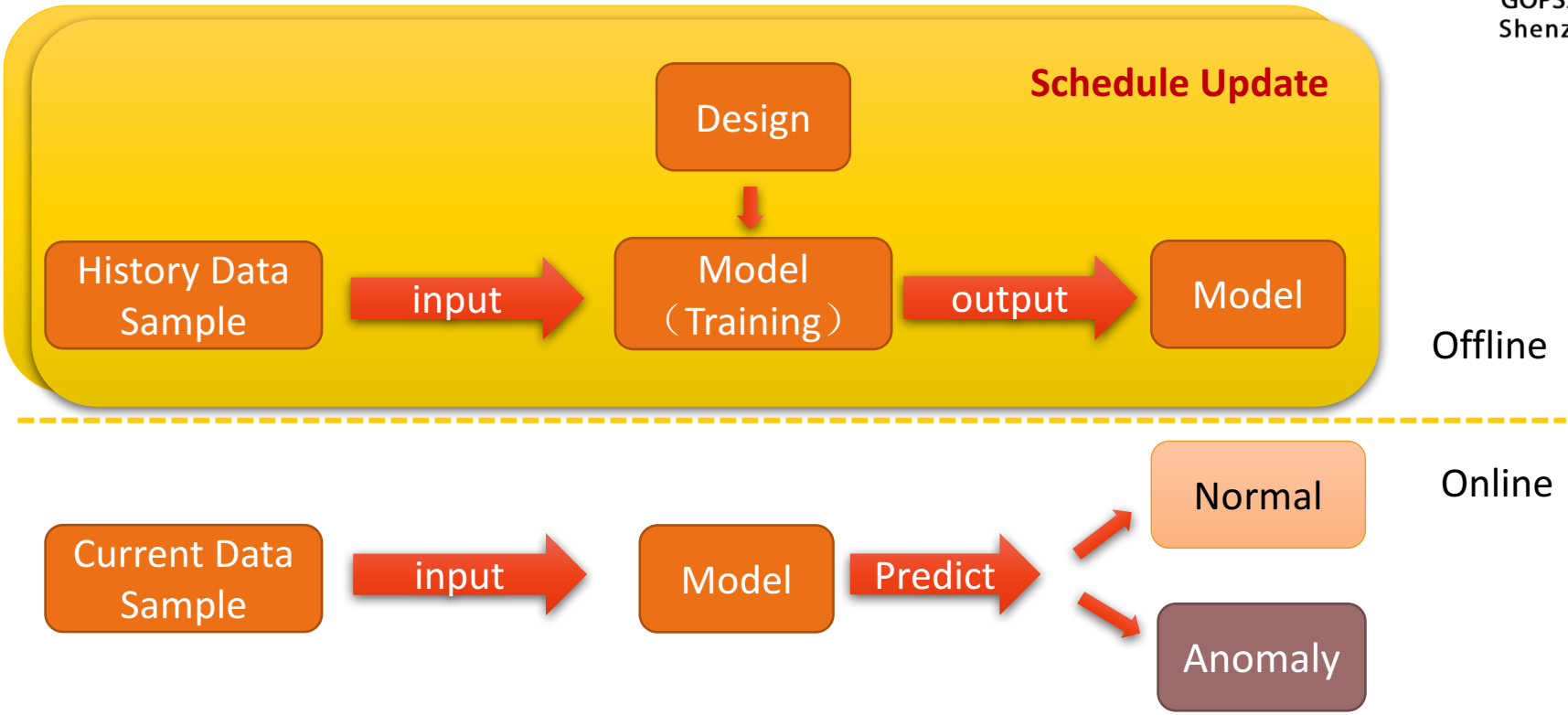
80%



20%

目录

机器学习架构


Schedule Update
Design
History Data Sample
input
Model（Training）
output
Model
Offline
Online
Current Data Sample
input
Model
Predict
Normal
Anomaly

学习方式选择

有监督

- 正负样本比例： 1 : 1
- 人工标注
- 有效的Boosting

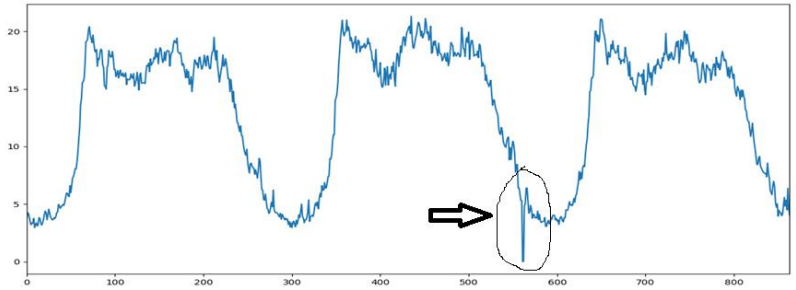
无监督

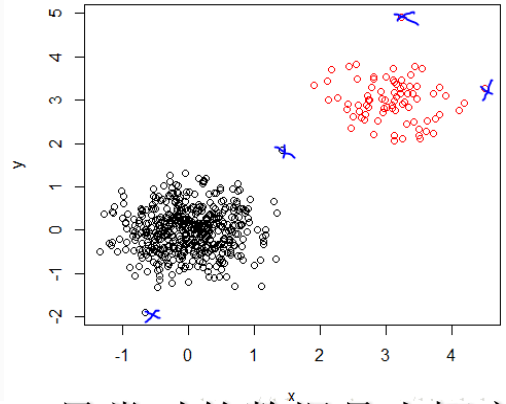
- 不必考虑正负样本比例
- 无需标注，自动学习信息
- 手工调参优化

特征提取

GOPS2018
Shenzhen

目标：





异常时一定有波动！　不符合历史波动规律！　异常时的数据是小概率事件！

特征向量：[[归一化流量大小(zs), 环比振幅(ca)]]

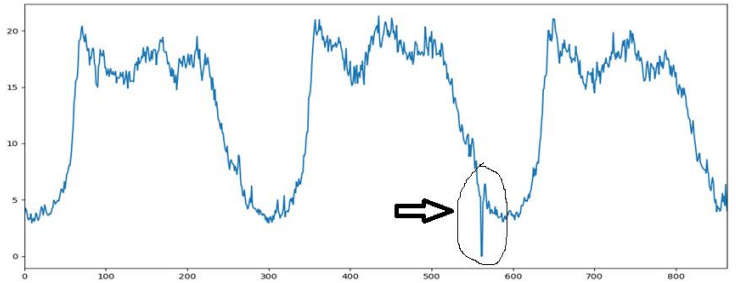
聚类经典：K-Means

Algorithm 1: K-Means Algorithm

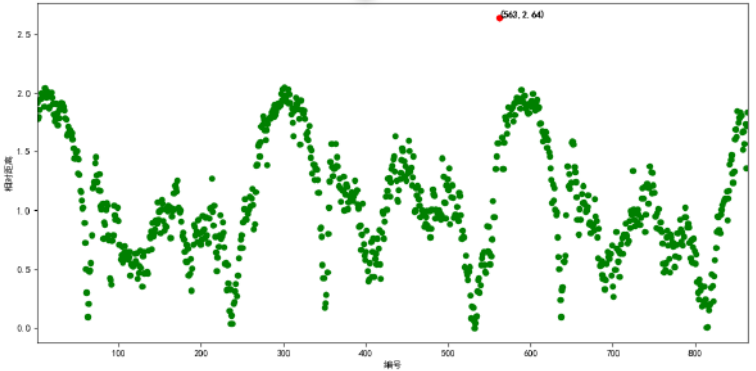
Input: $E = \{e_1, e_2, \ldots, e_n\}$ (set of entities to be clustered)
k (number of clusters)
$MaxIters$ (limit of iterations)

Output: $C = \{c_1, c_2, \ldots, c_k\}$ (set of cluster centroids)
$L = \{l(e) \mid e = 1, 2, \ldots, n\}$ (set of cluster labels of E)

```
foreach c_i ∈ C do
    c_i ← e_j ∈ E (e.g. random selection)
end
foreach e_i ∈ E do
    l(e_i) ← argminDistance(e_i, c_j) j ∈ {1...k}
end

changed ← false;
iter ← 0;
repeat
    foreach c_i ∈ C do
        UpdateCluster(c_i);
    end
    foreach e_i ∈ E do
        minDist ← argminDistance(e_i, c_j) j ∈ {1...k};
        if minDist ≠ l(e_i) then
            l(e_i) ← minDist;
            changed ← true;
        end
    end
    iter++;
until changed = true and iter ≤ MaxIters ;
```



k = 2, threshold = 2.4

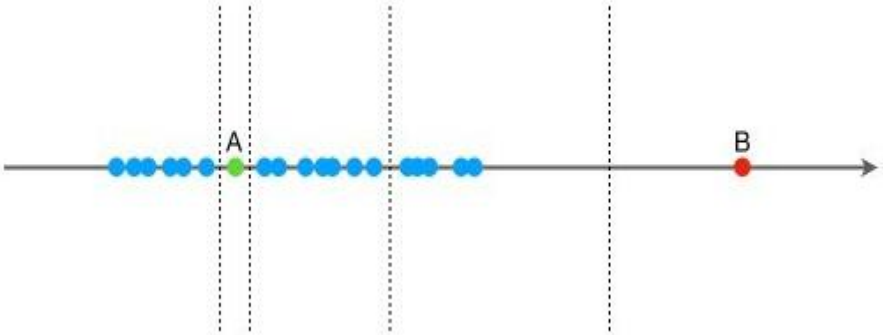


GOPS2018
Shenzhen

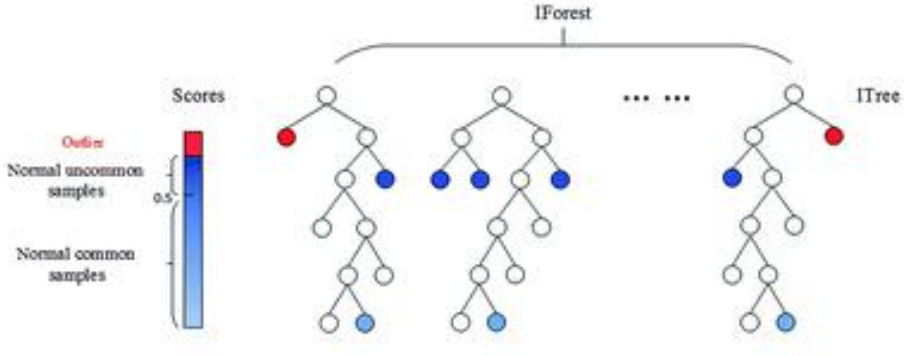
异常检测中的黑马: Isolation Forest

GOPS2018 Shenzhen

算法来自周志华老师在2011年的文献：***Isolation-based Anomaly Detection***



思想



Model

K-Means vs IForest

GOPS2018
Shenzhen

	K-Means	IForest
特征数量要求	相对较多	无需太多特征
训练性能	采用循环距离计算，训练效率相对较低	采用二叉树，训练效率快
预测性能	比较质心点距离--较快	遍历树后综合评分-较快
分类设定	需要对样本提前设定类别数量	不需要
易用性	需要删除离群点--差	自动规避异常--好

我们最终的选择： **IForest**

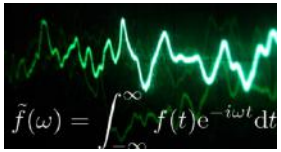
模型训练细节

- 模型设计
  2 Model(in/out)/Port
- 样本选取
  Last 7 days
- 窗口大小设定
  10 Minutes
- 模型更新
  Daily update

IForest
- 最大估计数= 7
- 最大样本数= 256
- 数据剔除率 = 0.01

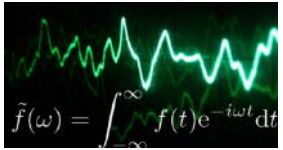
效果

算法与机器学习相结合

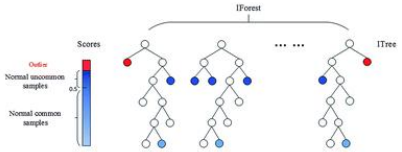


多算法仲裁

准确率：79%



多算法仲裁



模型仲裁

准确率：98.5%

目录

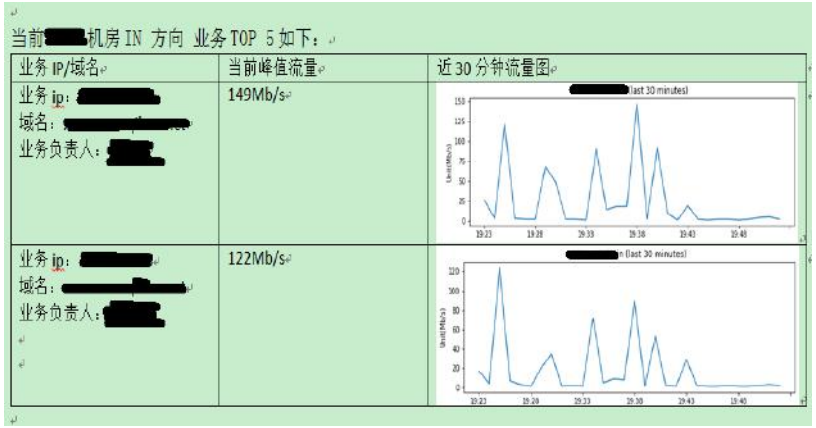
TopN数据参考

GOPS2018
Shenzhen

IDC流量TopN关联展现

基于之前的积累，通过API获取IDC当前 IN/OUT方向业务TOPN数据

异常出现



曲线相关性定位业务

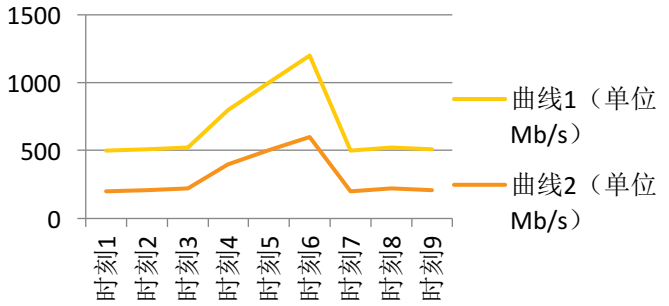
GOPS2018 Shenzhen

Pearson相关系数

用于分析两个连续性变量之间的关系，公式：

$$r = \frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2\sum_{i=1}^{n}(y_i-\bar{y})^2}}$$

取值范围	相关程度
|r| ≤ 0.3	不存在线性相关
0.3 ≤ |r| ≤ 0.5	低度线性相关
0.5 ≤ |r| ≤ 0.8	显著线性相关
|r| > 0.8	高度线性相关

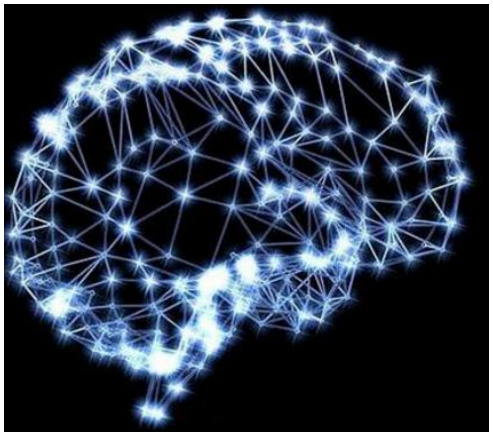



```
import numpy as np

a = np.array([[500,510,520,800,1000,1200,500,523,511],
              [200,210,220,400,500,600,200,220,211]])

print np.corrcoef(a)
```

```
[[ 1.          0.99829181]
 [ 0.99829181  1.        ]]
[Finished in 0.2s]
```

下一步要做的事情

- 关联分析
- 根因分析



故障预测

故障根本原因

自动启动预案

故障自愈

Thanks

高效运维社区

开放运维联盟

荣誉出品

想第一时间看到高效运维社区的新动态吗？

高维在线