



Anomaly Detection

Softnix Technology
Chakrit Phain



Agenda Day 1

- Introduction to Anomaly Detection
- Anomaly Detection with Distribution
- Anomaly Detection with Boxplot
- Anomaly Detection with DB-Scan
- Anomaly Detection with Local Outlier Factor



Agenda Day 2

- Anomaly Detection and Supervised Learning
- Neural Network & Deep Learning
- Anomaly Detection with LSTM & GRU
- Anomaly Detection with Autoencoder
- Anomaly Detection with Variational Autoencoder

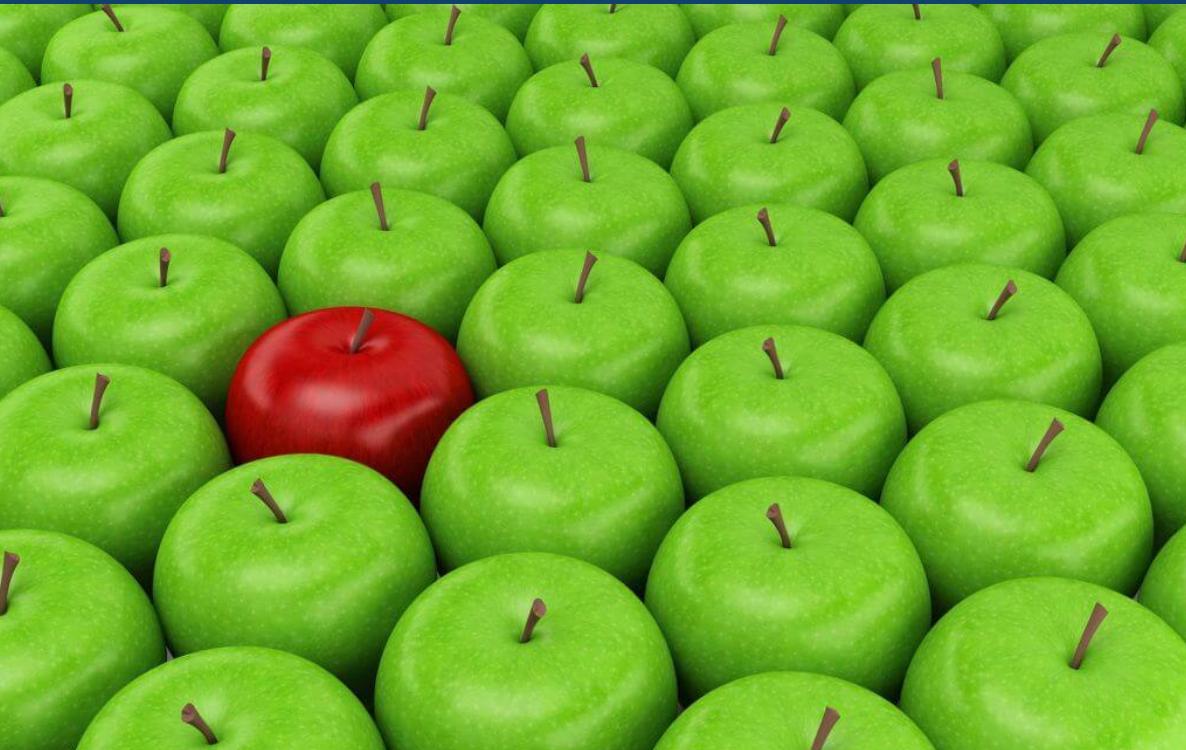


Anomaly Detection

Day 1



Introduction to Anomaly Detection

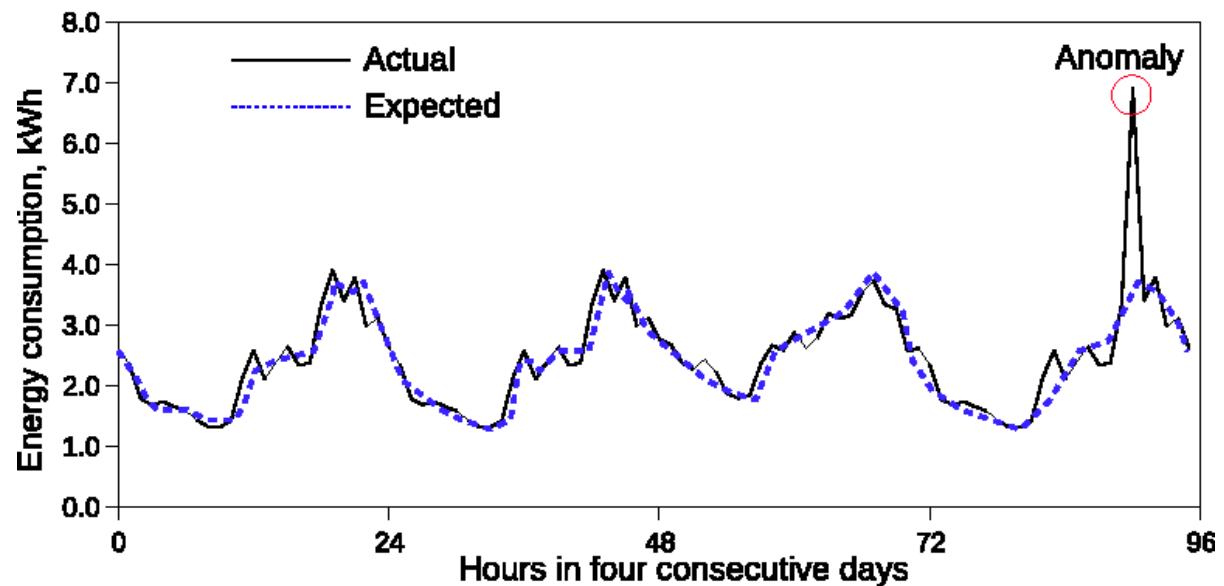


Introduction

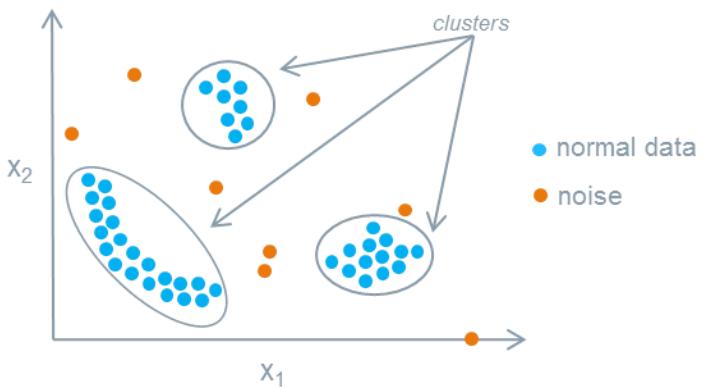
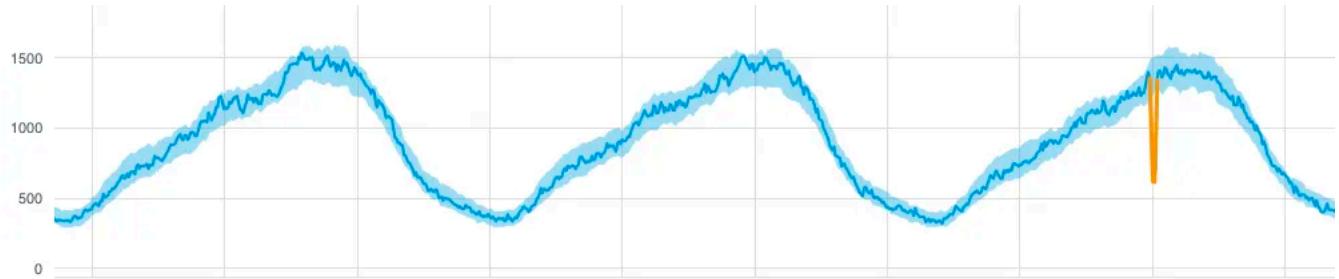


Introduction

In [data mining](#), **anomaly detection** (also **outlier detection**^[1]) is the identification of rare items, events or observations which raise suspicions by differing significantly from the majority of the data.



Introduction



<https://www.anodot.com/blog/what-is-anomaly-detection/>

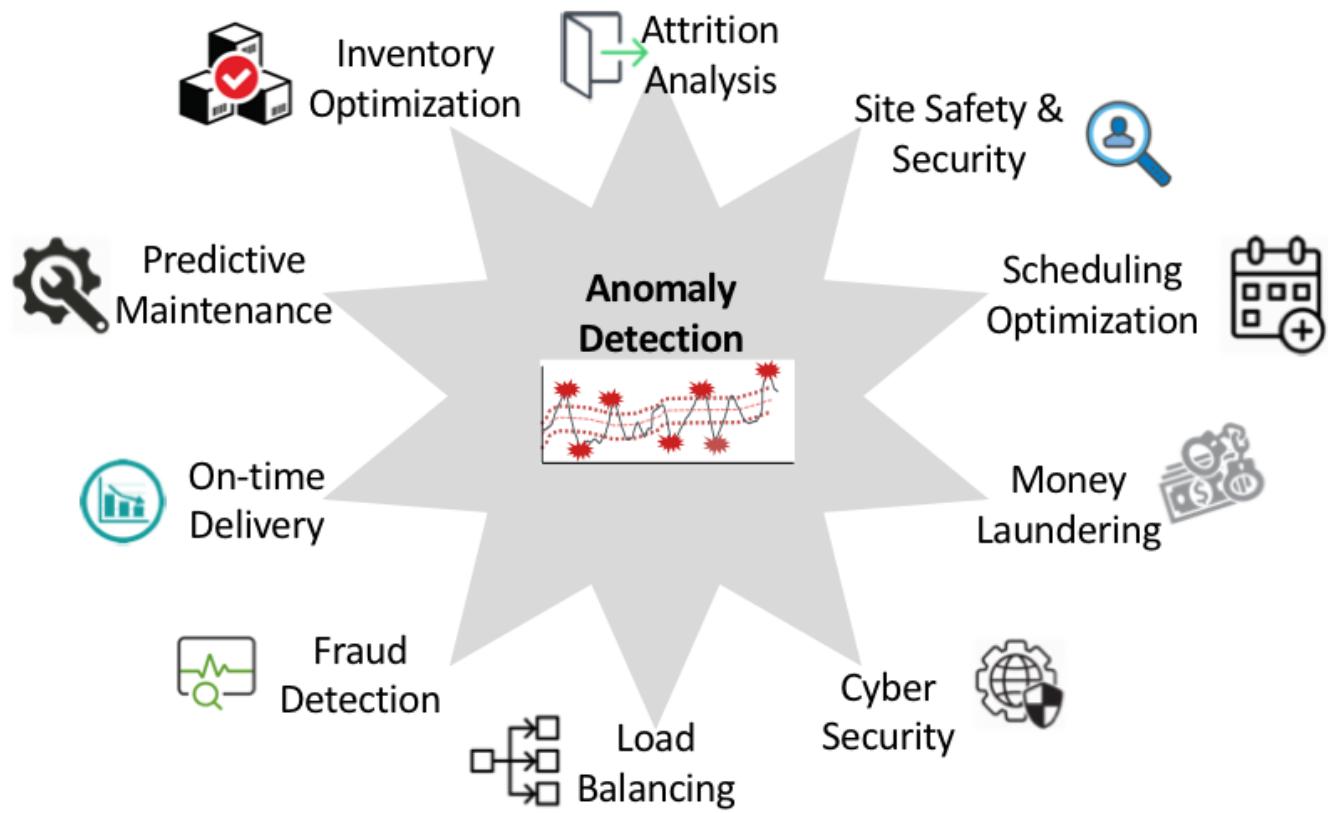
<https://developer.mindsphere.io/apis/analytics-anomalydetection/api-anomalydetection-overview.html>



Applications of Anomaly Detection in the Business world

- Intrusion detection
 - Attacks on computer systems and computer networks.
- Fraud detection
 - The purchasing behavior of some one who steals a credit card is probably different from that of the original owner.
- Ecosystem Disturbance.
 - Hurricanes , floods , heat waves ... etc
- Medicine.
 - Unusual symptoms or test result may indicate potential health problem.

Applications of Anomaly Detection in the Business world

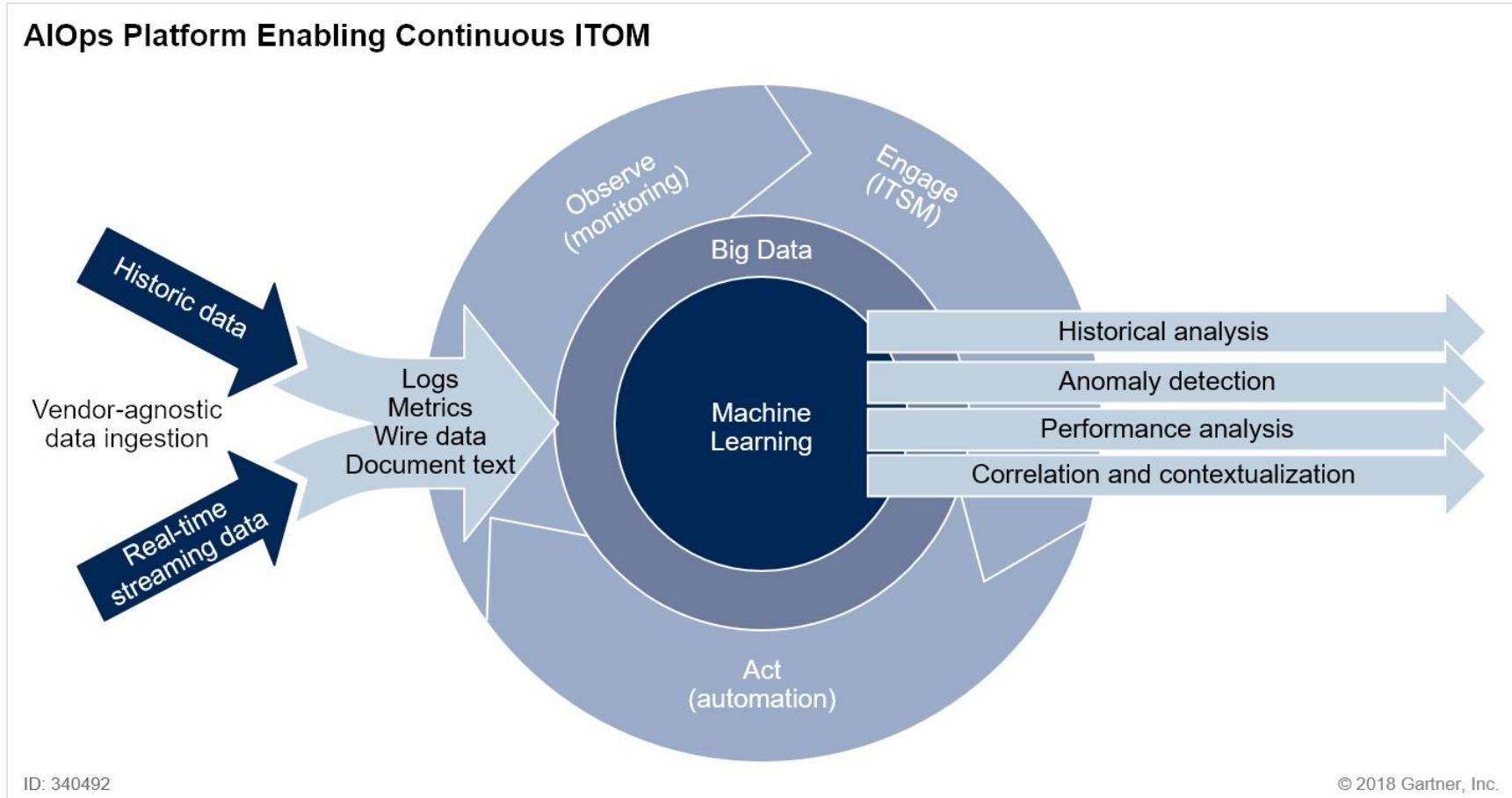




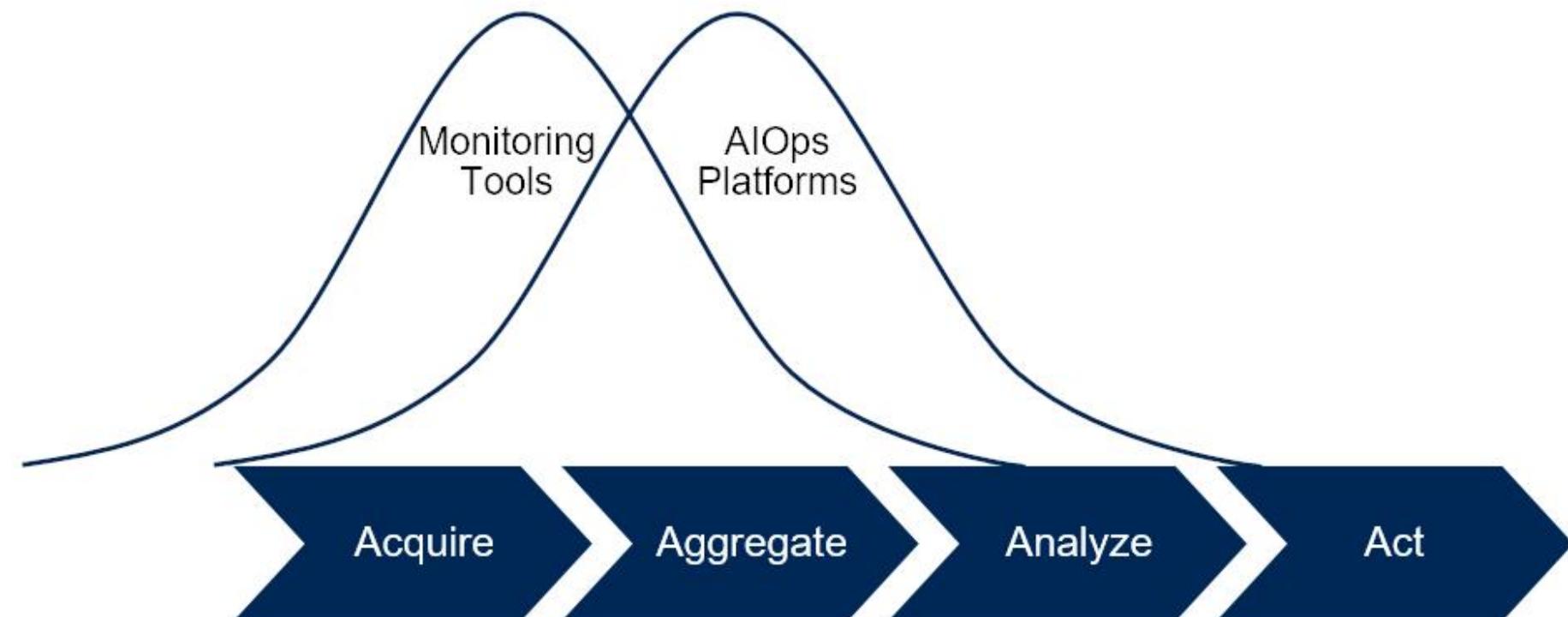
AIOps



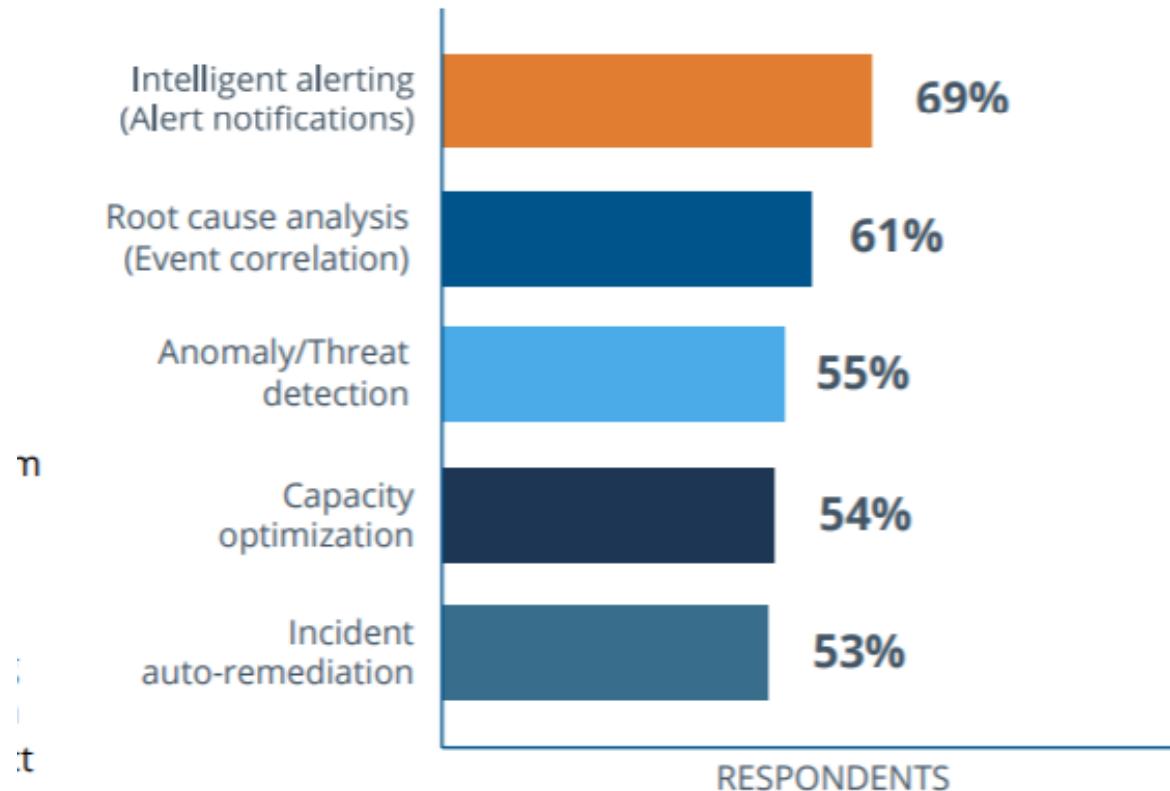
AIOps



Four Stages of Monitoring



HOW IS YOUR TEAM CURRENTLY USING AIOPS TOOLS?





Graph Database



Fraud Detection for Graph analytic

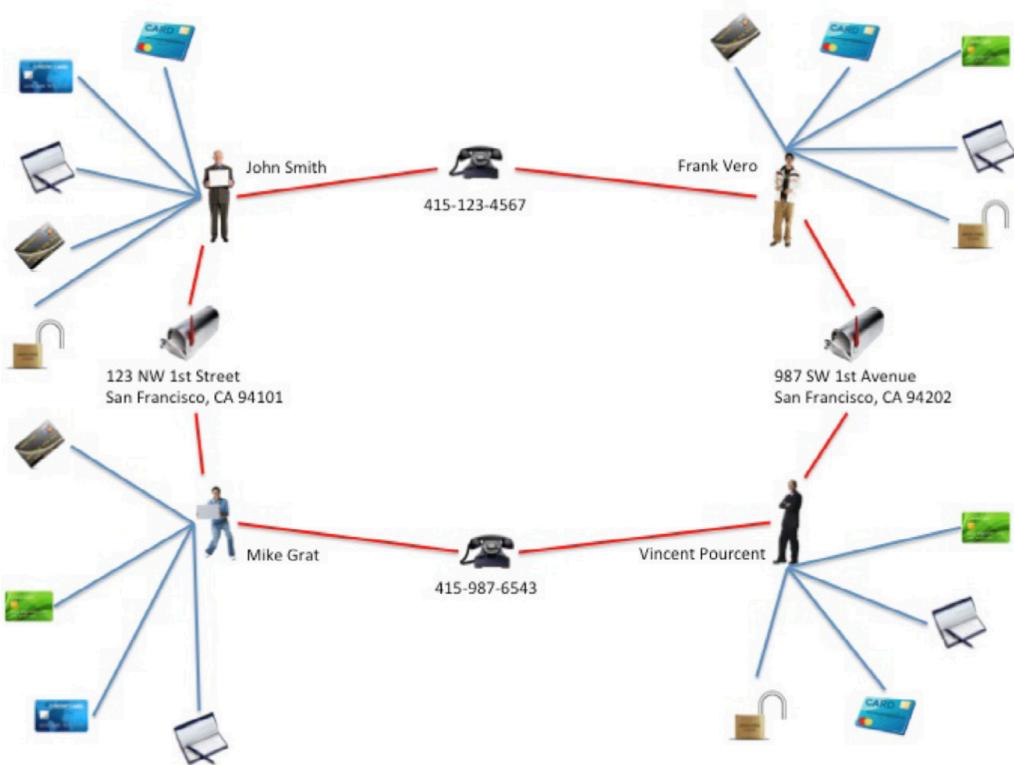


Diagram 1: 2 people sharing 2 pieces of data and creating 4 synthetic identities

Insurance Fraud

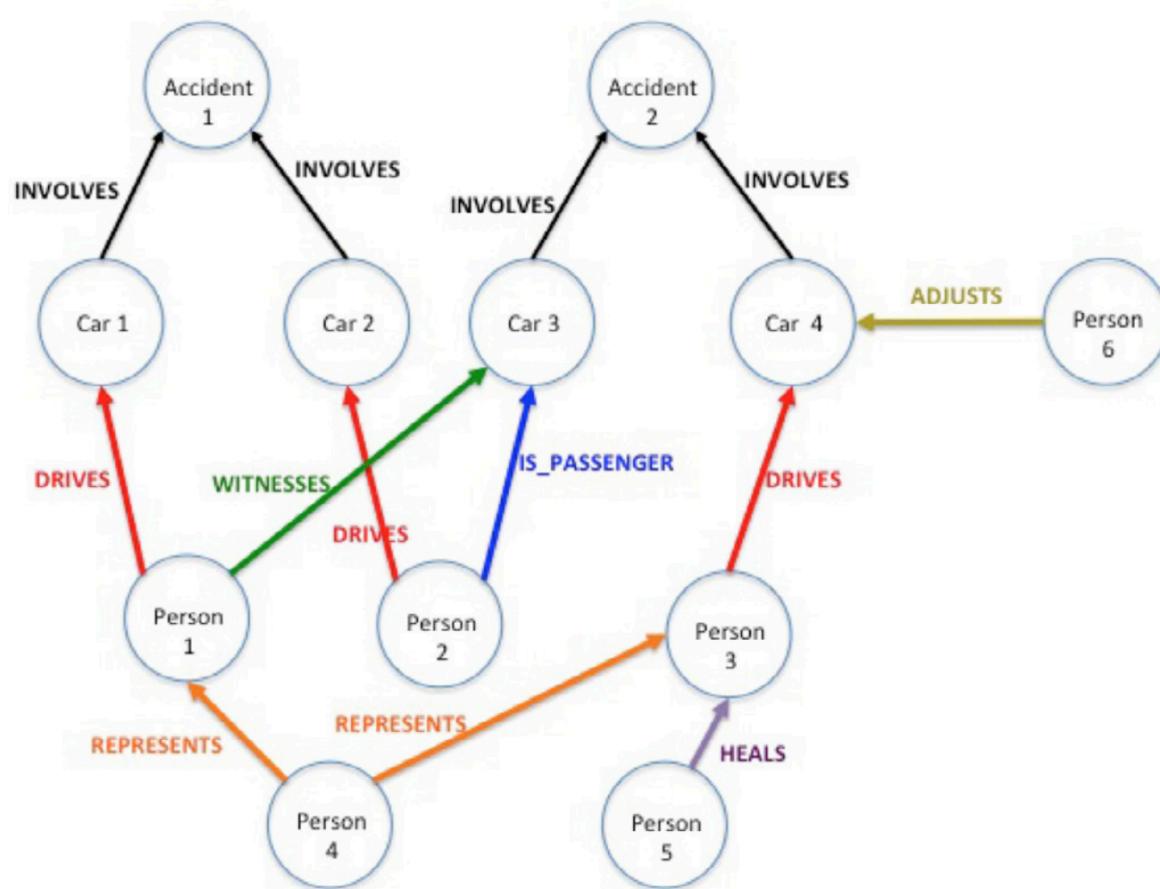


Diagram 7: A Graph Representation of Insurance Fraud

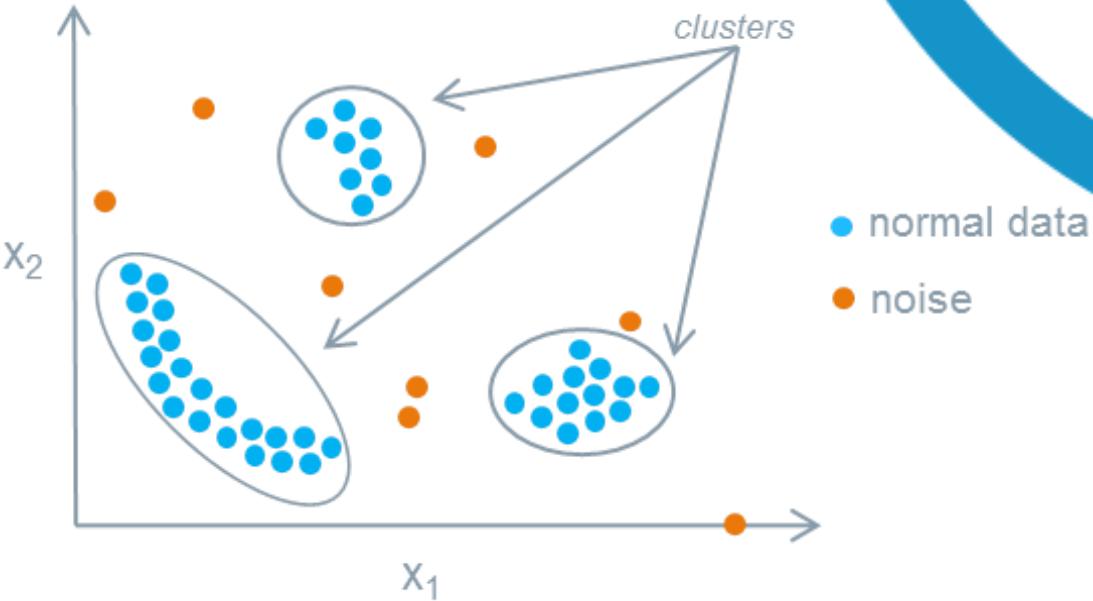


Type of Anomaly



Anomalies can be broadly categorized as

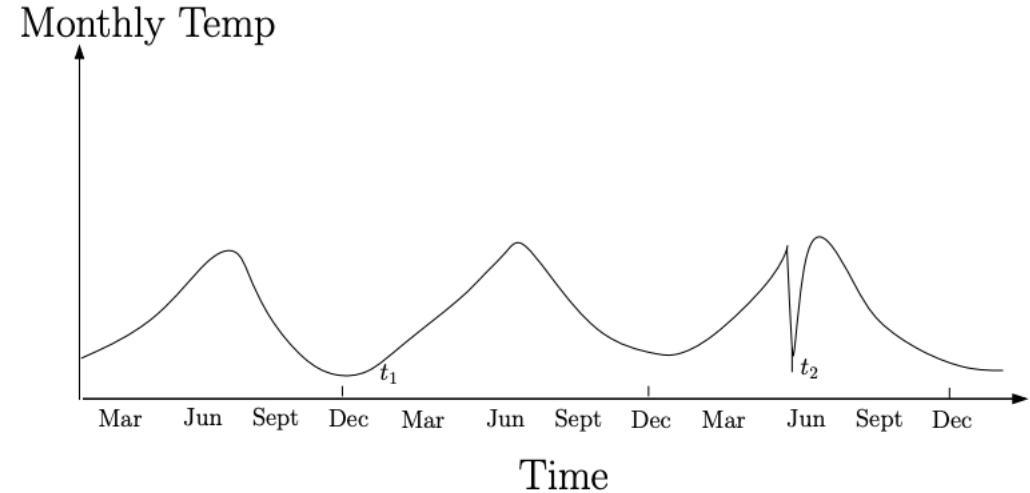
1. Point Anomalies
2. Contextual Anomalies
3. Collective Anomalies



If an individual data instance can be considered as anomalous with respect to the rest of data, then the instance is termed as a point anomaly.

Anomalies can be broadly categorized as

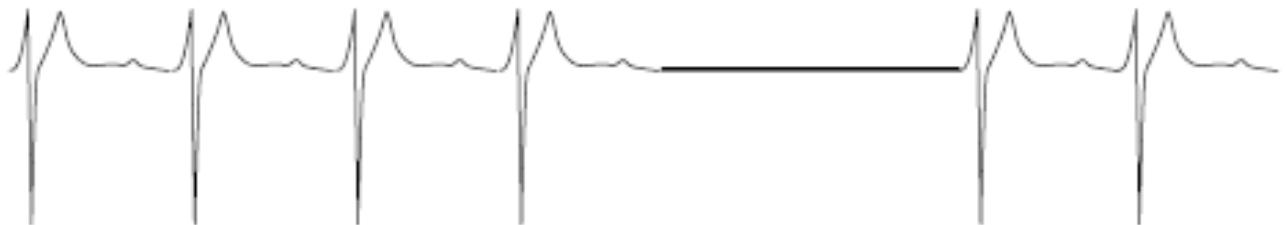
1. Point Anomalies
2. **Contextual Anomalies**
3. Collective Anomalies



If a data instance is anomalous in a specific context (but not otherwise), then it is termed as a contextual anomaly

Anomalies can be broadly categorized as

1. Point Anomalies
2. Contextual Anomalies
3. **Collective Anomalies**



If a collection of related data instances is anomalous with respect to the entire data set, it is termed as a collective anomaly



Machine Learning-Based Approaches for Anomaly Detection

- 
- 1. Clustering-Based Anomaly Detection
 - 2. Density-based anomaly detection
 - 3. Support Vector Machine-Based Anomaly Detection
 - 4. Supervised Deep Anomaly Detection
 - 5. Semi-supervised deep anomaly detection
 - 6. One-Class Neural Networks (OC-NN)



How to Identify Outliers in your Data

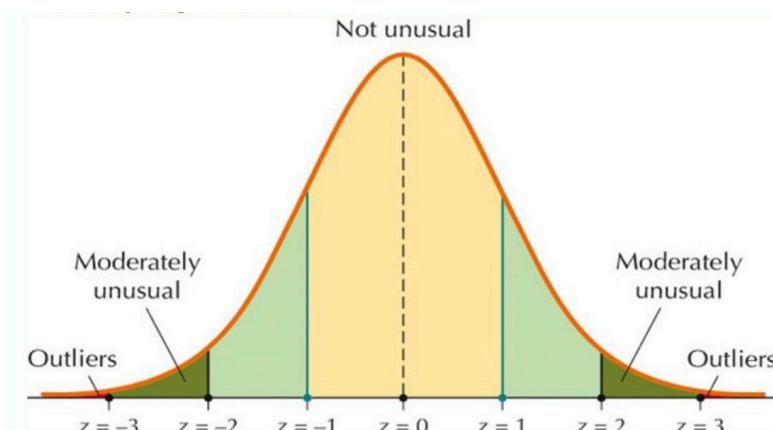
- 1. Extreme Value Analysis**
- 2. Proximity Methods**
- 3. Projection Methods**
- 4. Methods Robust to Outliers**



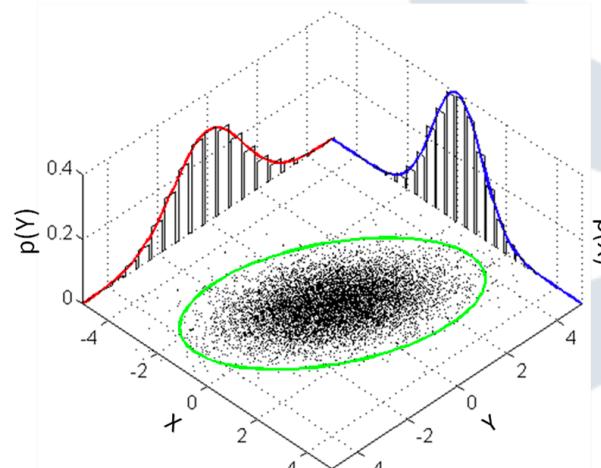
Welcome! I'm **Jason Brownlee PhD** and I help developers get results with machine learning.

[Read More](#)

1. Extreme Value Analysis
2. Proximity Methods
3. Projection Methods
4. Methods Robust to Outliers



1. Focus on univariate methods
2. Visualize the data using scatterplots, histograms and box and whisker plots and look for extreme values
3. Assume a distribution (Gaussian) and look for values more than 2 or 3 standard deviations from the mean or 1.5 times from the first or third quartile
4. Filter out outliers candidate from training dataset and assess your models performance



<https://machinelearningmastery.com/how-to-identify-outliers-in-your-data/>

https://en.wikipedia.org/wiki/Multivariate_normal_distribution

How to Identify Outliers in your Data

1. Extreme Value Analysis
2. Proximity Methods
3. Projection Methods
4. Methods Robust to Outliers

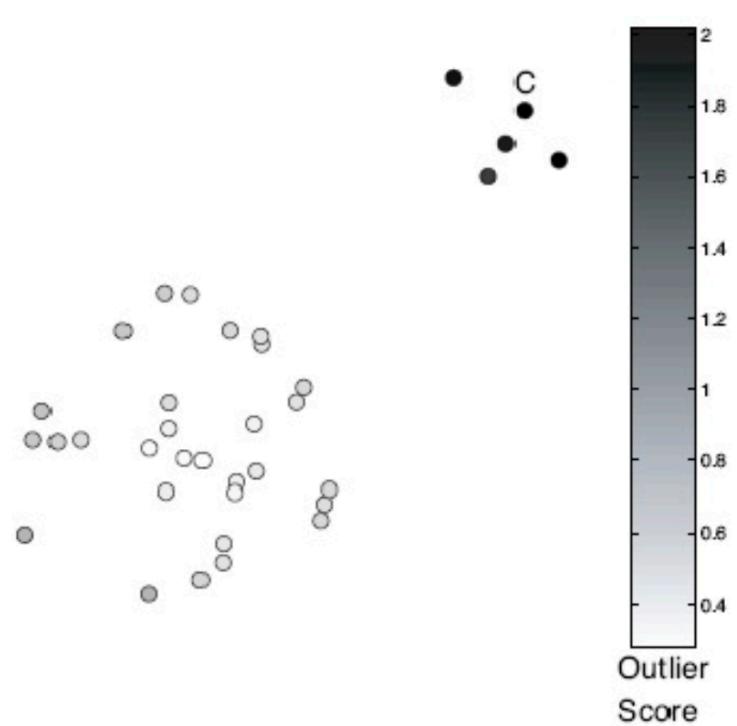
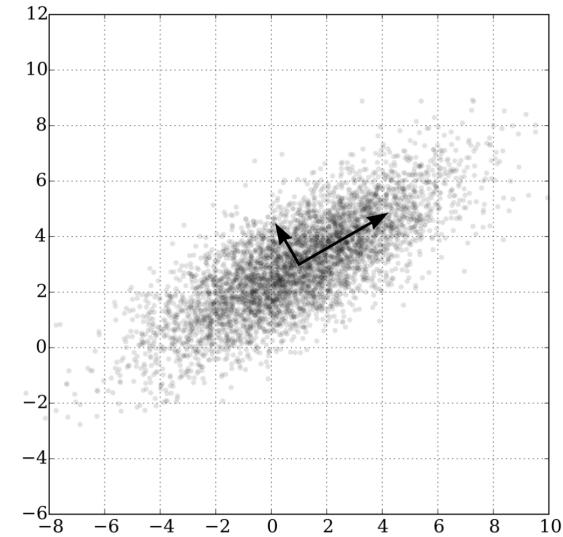
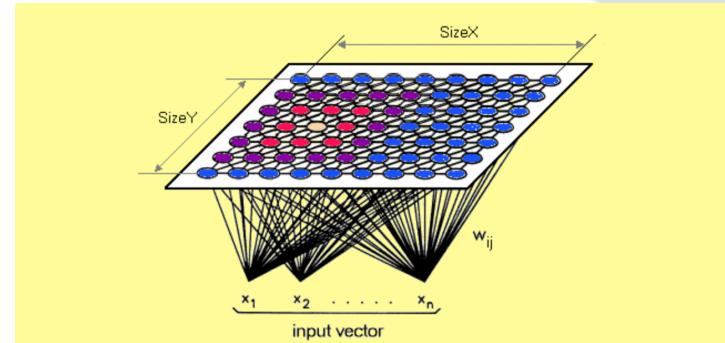


Figure 10.6. Outlier score based on distance to the fifth nearest neighbor. A small cluster becomes an outlier.

How to Identify Outliers in your Data

1. Extreme Value Analysis
2. Proximity Methods
3. **Projection Methods**
4. Methods Robust to Outliers

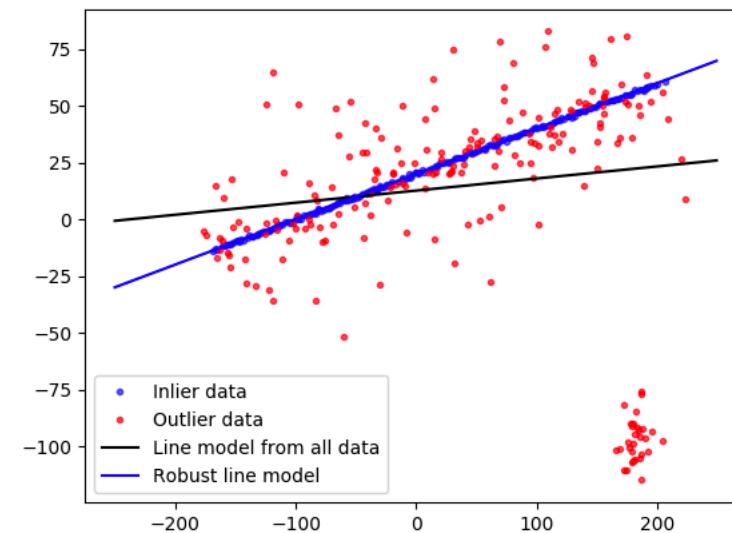


1. Extreme Value Analysis
2. Proximity Methods
3. Projection Methods
4. Methods Robust to Outliers

$$MAE = \frac{\sum_{i=1}^n |y_i - \hat{y}_i|}{n}$$

$$minkowski_error = \frac{\sum (outputs - targets)^{minkowski_parameter}}{instances_number}$$

$$mean_squared_error = \frac{\sum (outputs - targets)^2}{instances_number}$$



Typology of Anomalies

Cardinality of Relationship	Types of data		
	Continuous attributes	Categorical attributes	Mixed attributes
Univariate Described by individual attributes (independence)	Type I Extreme value anomaly	Type II Rare class anomaly	Type III Simple mixed data anomaly
Multivariate Described by multidimensionality (dependence)	Type IV Multidimensional numerical anomaly	Type V Multidimensional rare class anomaly	Type VI Multidimensional mixed data anomaly

From: Foorthuis (2018), 'A Typology of Data Anomalies', IPMU 2018.

Anomaly based IDS using Backpropagation Neural Network

- Vrushali D. Mane ME (Electronics) JNEC, Aurangabad
- S.N. Pawar Associate Professor JNEC, Aurangabad
- Neural Network detect 98% accuracy

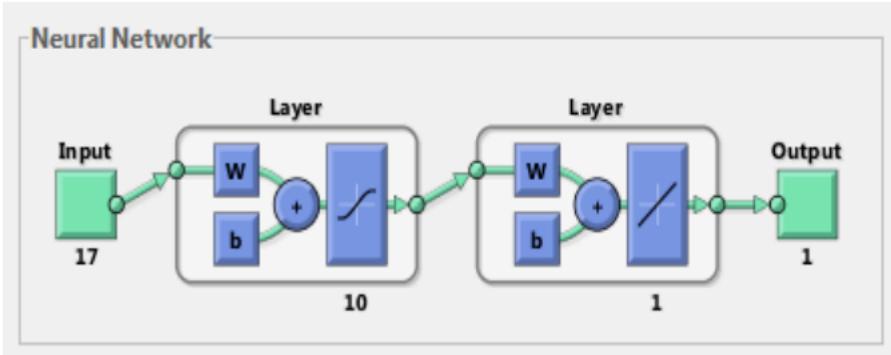
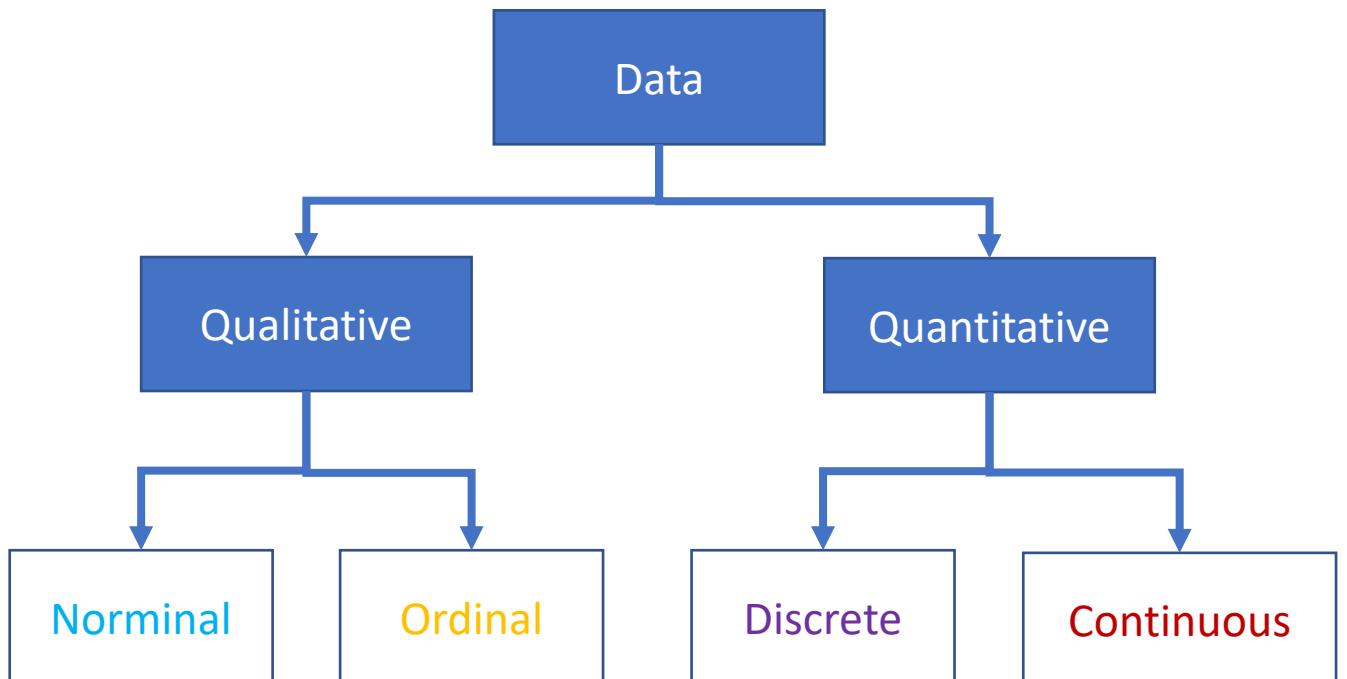


Fig 2: Neural network training model.

Type of Data



- ชัย , หญิง
- PC , Notebook , Tablet
- กาแฟ , น้ำเปล่า , น้ำส้ม
- กองแดง , เงิน , กองร้อน , อุ่น , เย็น
- ปานกลาง , สวาย , สวยมาก
- 1 , 2 , 3 , 4 , 5
- 1 , 1.1 , 1.5 , 1.6 , 2



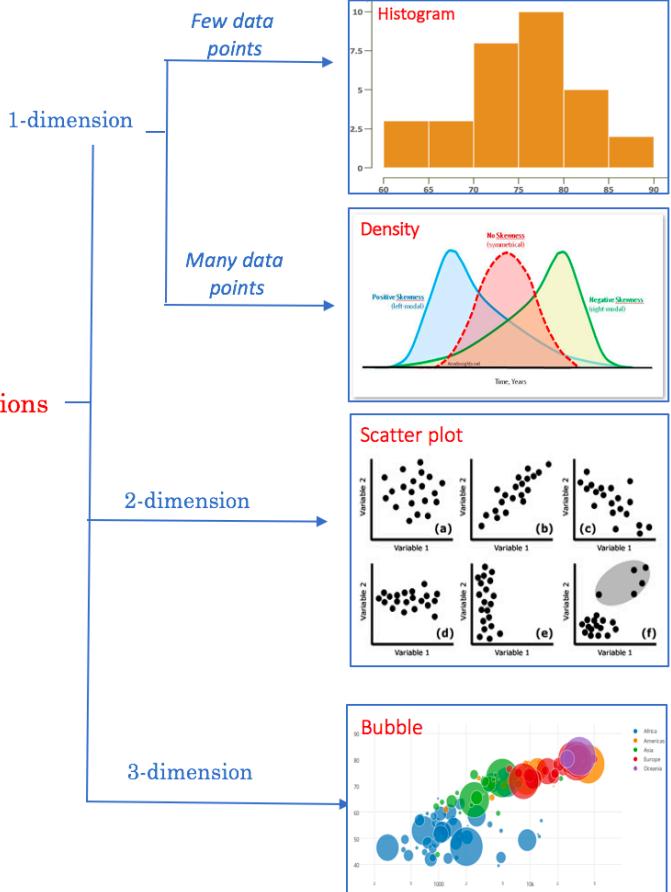


EDA Methods

Mean	Sum of all values Total number of values
Median	Middle value(when data are arranged in order)
Mode	Most common value
Variance	how far a set of numbers are spread out from mean
Interquartile range	divides a data set into quartiles.
Standard deviation	dispersion of a set of data from mean
Skewness	Measure of symmetry
Kurtosis	Kurtosis is a measure of “peakedness” relative to a Gaussian shape

EDA Methods

Descriptive statistics





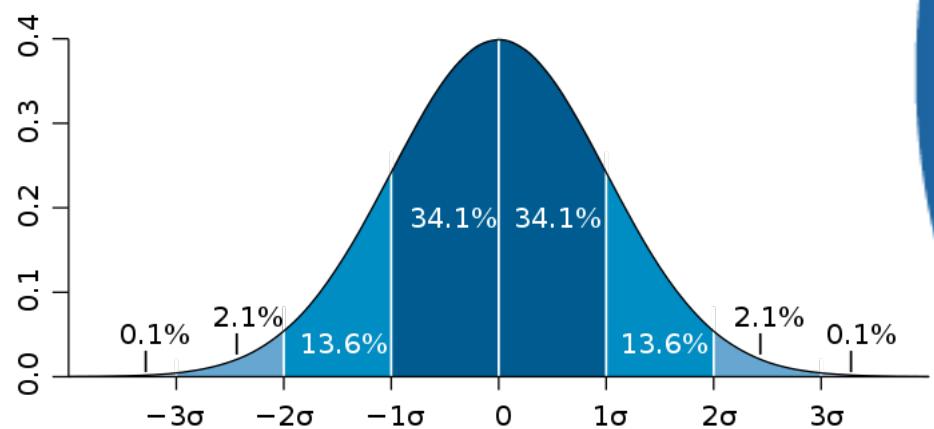
Anomaly Detection with Distribution



Normal Distribution

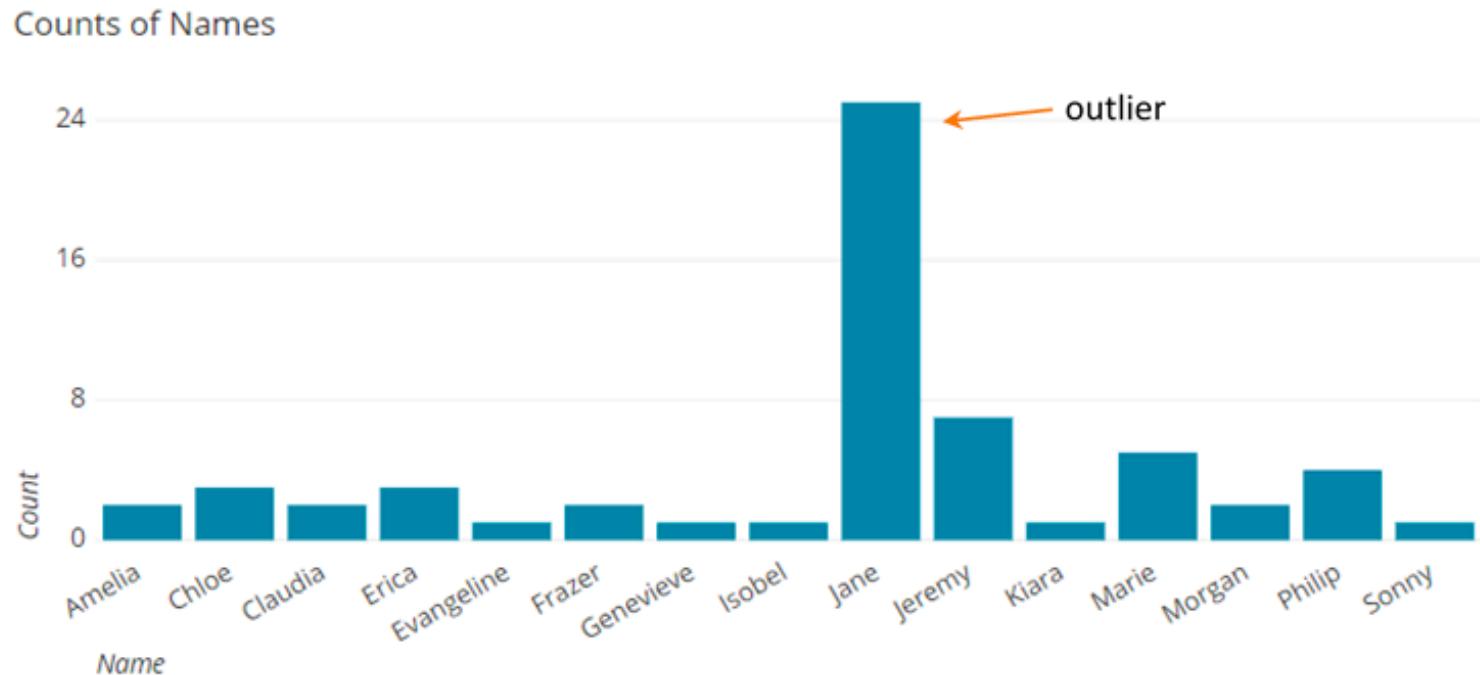
If a continuous random variable has a distribution with a graph that is symmetric and bell-shaped and can be described by the equation

$$\frac{1}{\sigma\sqrt{2\pi}}e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$



We say that it has a normal distribution

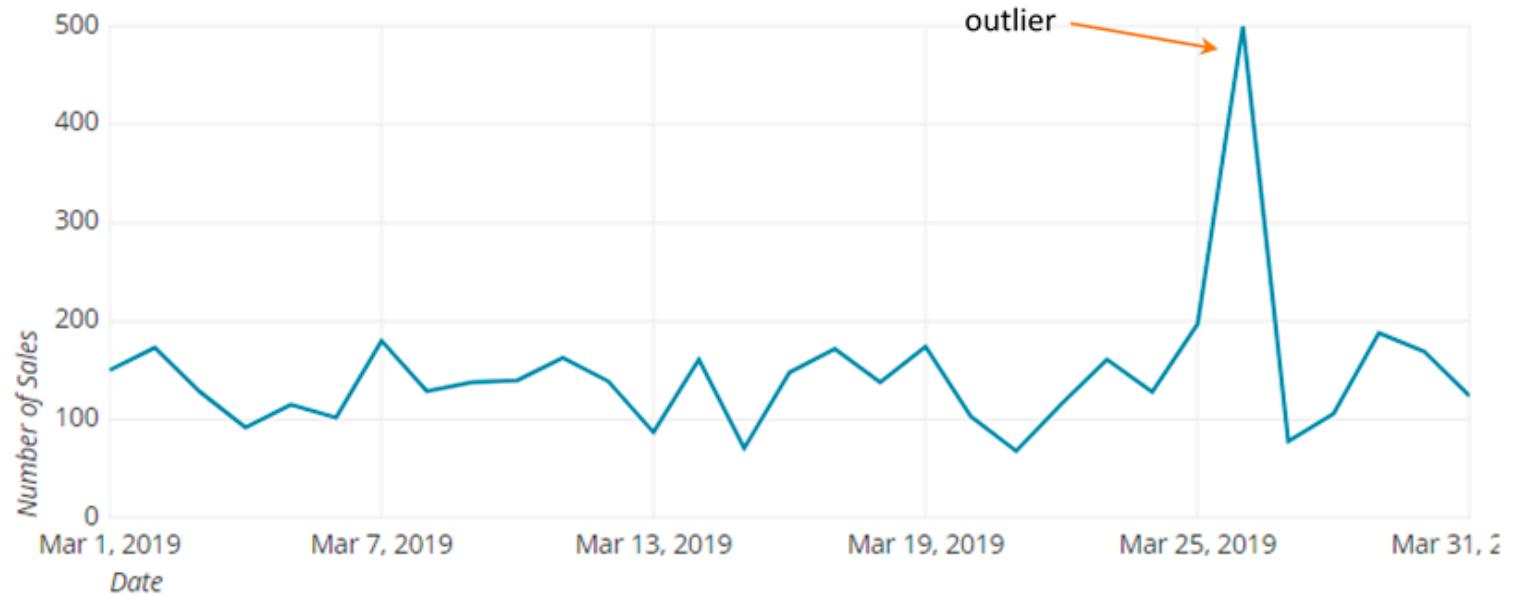
Anomaly Example





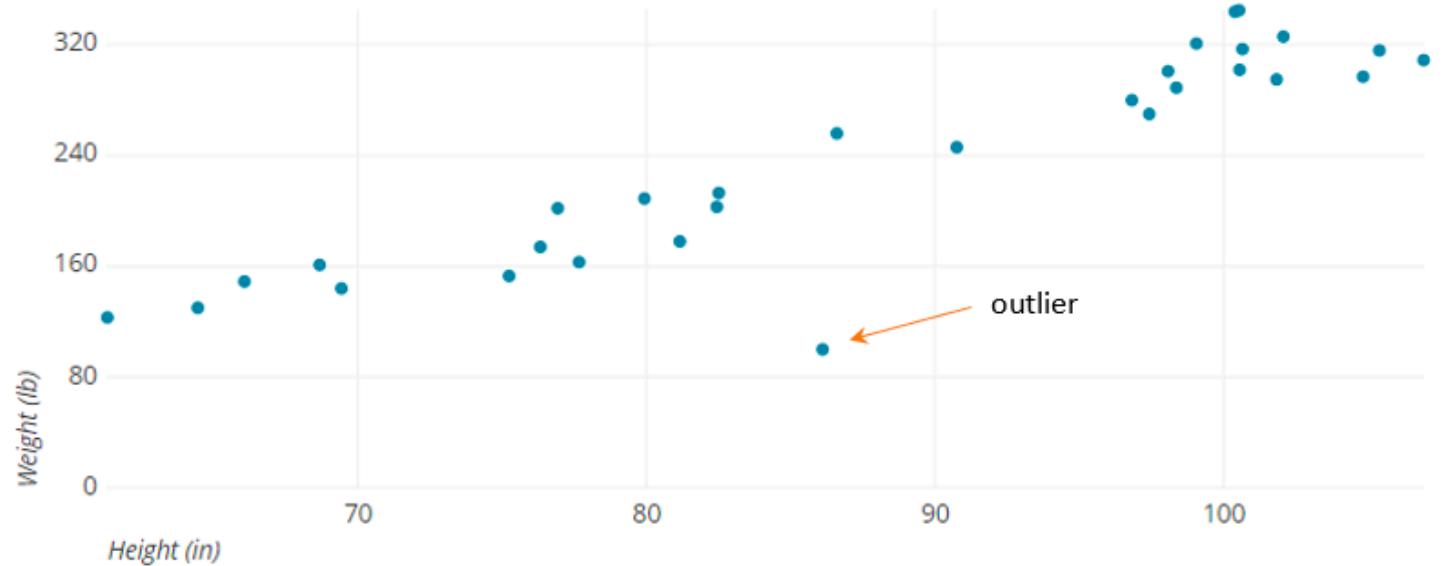
Anomaly Example

Sales per Day

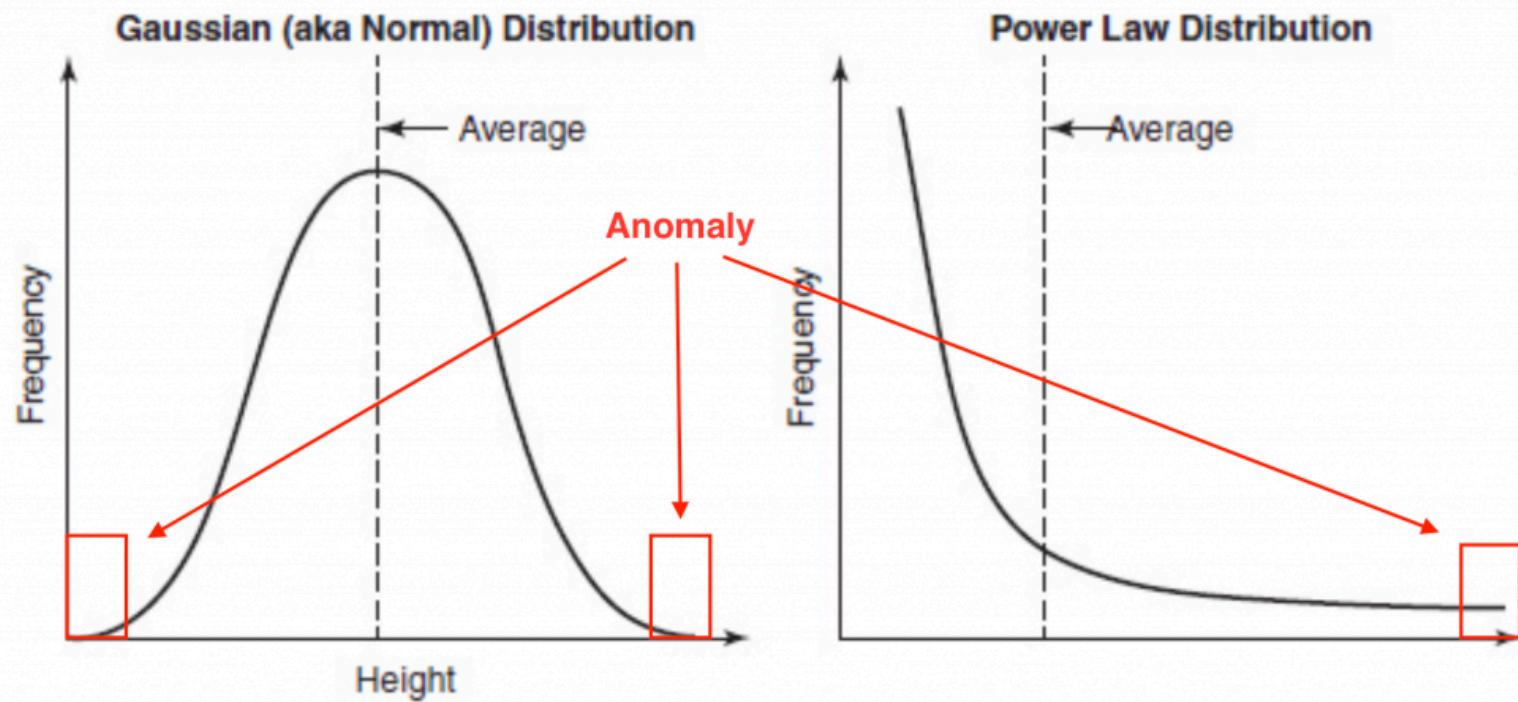


Anomaly Example

Weight v Height



Others Distributions





Anomaly Detection with Distribution Examples

451 lines (451 sloc) | 393 KB

[Open in Colab](#)

```
In [ ]: import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import requests
import re
import seaborn as sns
from scipy.stats import norm
```

ตัวอย่าง CPU

```
In [ ]: cpu = pd.read_csv("https://raw.githubusercontent.com/numenta/NAB/master/data/realAWScloudwatch/ec2_cpu_utilization_53ea38.csv")
```

```
In [ ]: f, axes = plt.subplots(1, 2, figsize=(10, 3))
sns.distplot(cpu.value,bins=30,fit=norm,ax=axes[0])
#สร้าง Threshold ที่ 3SD หรือประมาณ 99.7% ของ Data ทั้งหมด
threshold = cpu.value.mean() + 3*cpu.value.std()
axes[0].plot([u3sd,u3sd],[0,6])
axes[1].plot(cpu.value)
for i in range(len(cpu.value)):
    if cpu.value.values[i] > threshold:
        #Remark Threshold
        axes[1].plot(i,cpu.value.values[i],'ro')
```

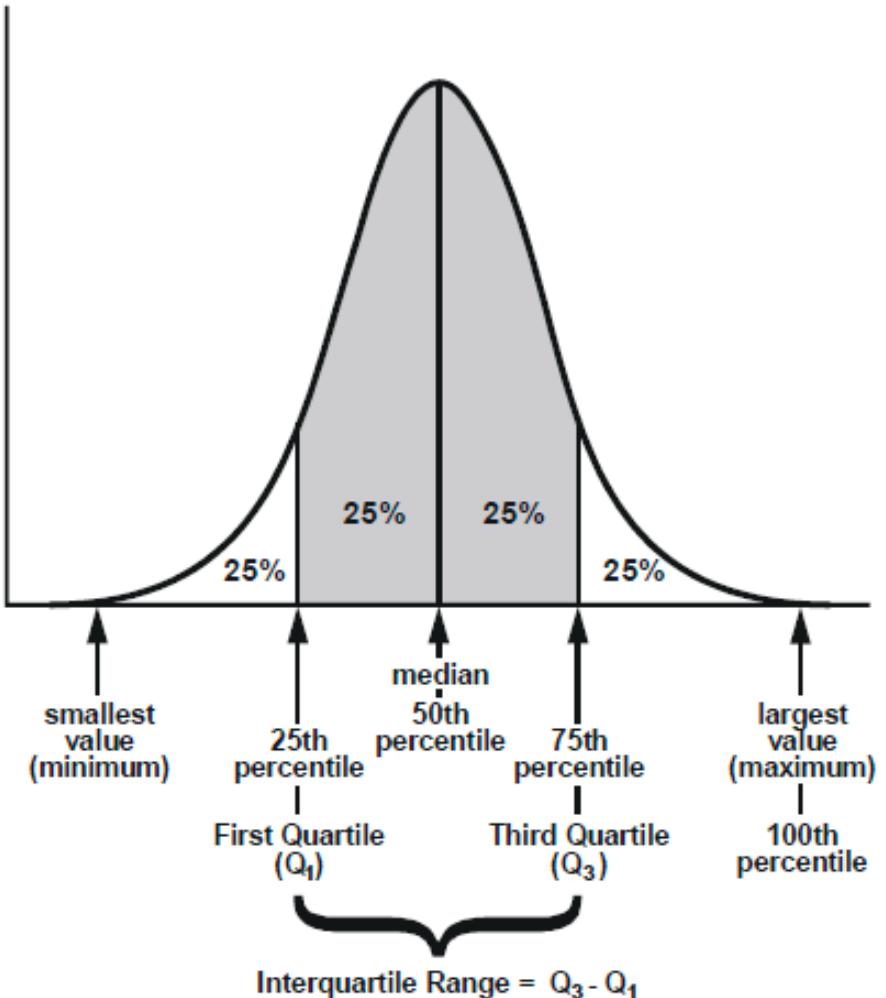
The figure consists of two side-by-side plots. The left plot is a histogram of CPU utilization values with a normal distribution curve overlaid. A vertical orange line marks the 3Sigma threshold at approximately 2.1. The right plot is a time-series plot of CPU utilization over 4000 samples. Red dots highlight several data points that fall outside the 3Sigma threshold, indicating anomalies.



Anomaly Detection with Boxplot

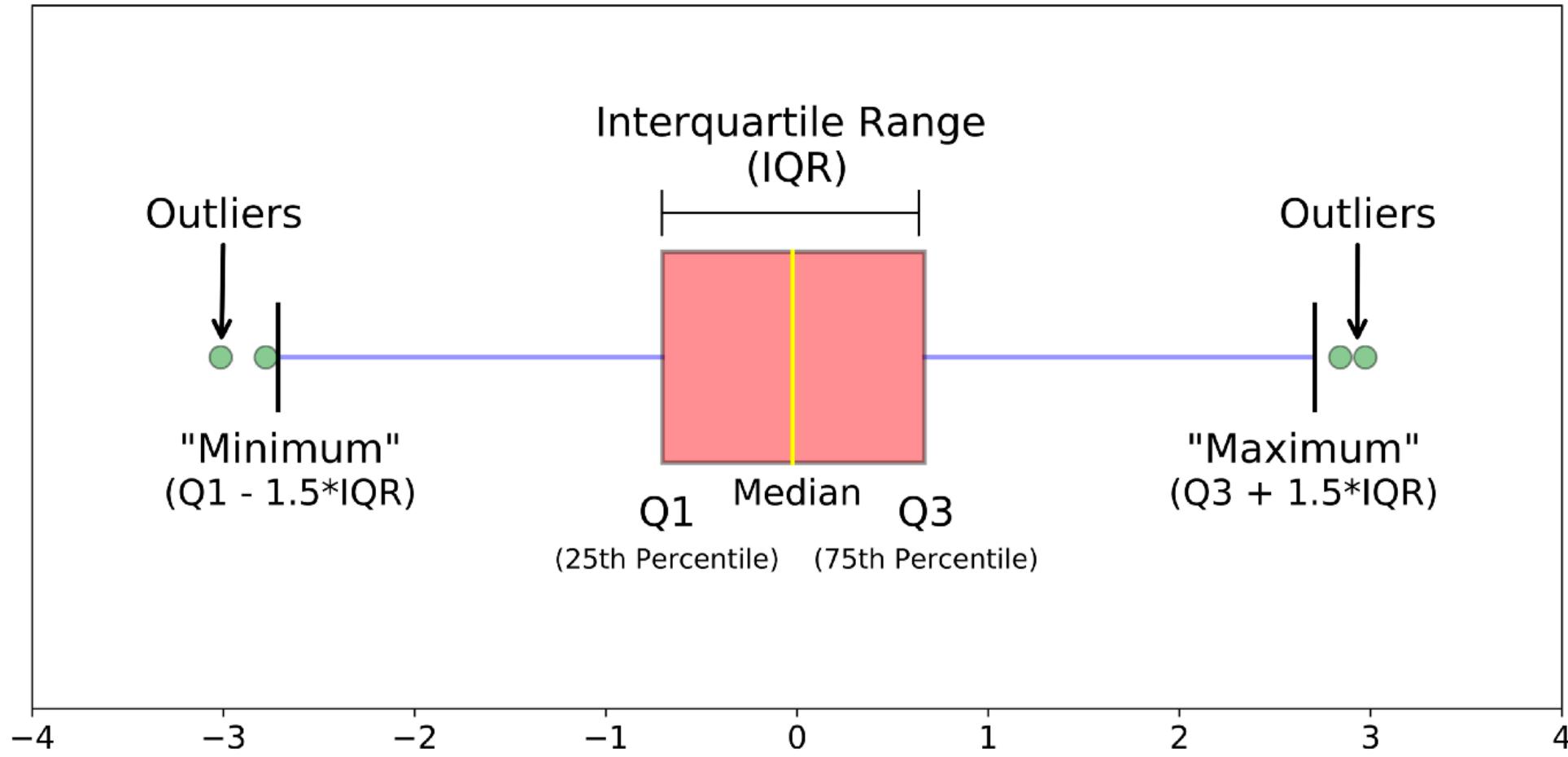


Understanding Boxplots





Understanding Boxplots



Understanding Boxplots

Math Expression

$$\int_{-.6745}^{.6745} \frac{1}{\sqrt{2\pi}} e^{-x^2/2} dx$$

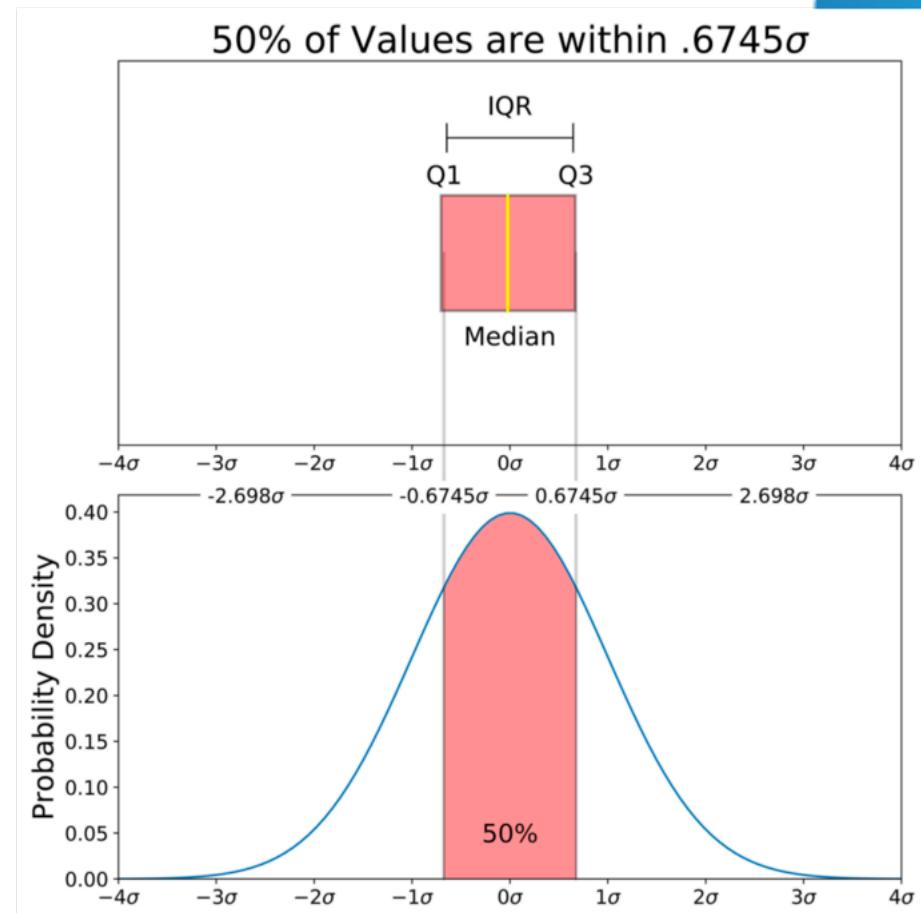
Code to Integrate

```
# Make a PDF for the normal distribution a function
def normalProbabilityDensity(x):
    constant = 1.0 / np.sqrt(2*np.pi)
    return(constant * np.exp((-x**2) / 2.0))

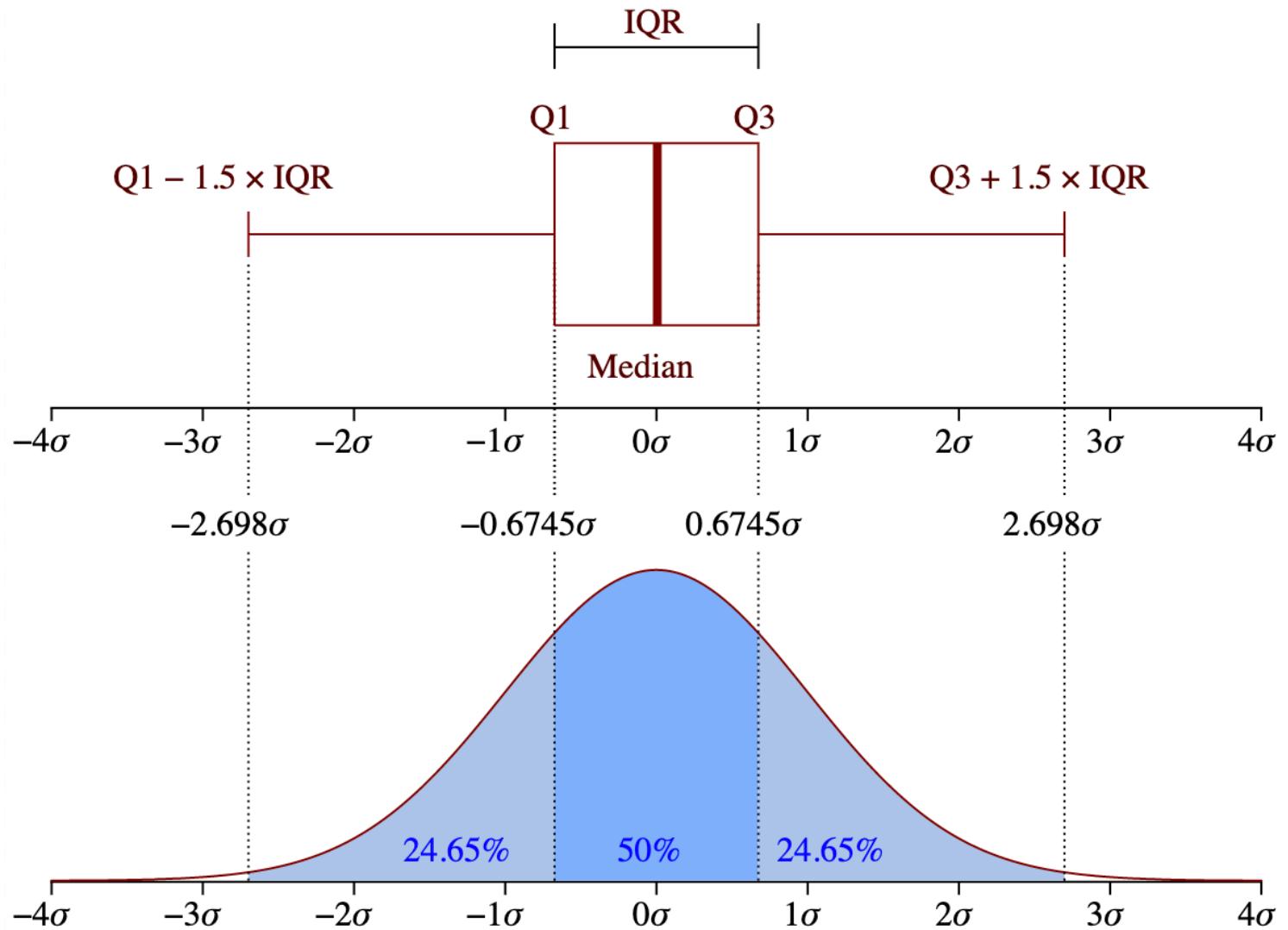
# Integrate PDF from -.6745 to .6745
result_50p, _ = quad(normalProbabilityDensity,
                     -.6745,
                     .6745,
                     limit = 1000)

print(result_50p)
```

0.500006514273



Boxplot on Normal Distribution





Boxplot on Normal Distribution

324 lines (324 sloc) | 11.6 KB

In [1]: `import numpy as np
from numpy.random import randn
import matplotlib.pyplot as plt`

In [14]: `data = np.array([4, 4, 4, 4, 4, 8, 7, 4, 4, 2, 6, 3, 5, 8, 6, 6, 4, 3, 1, 4, 4, 9, 2, 7, 4, 7, 6, 5, 3, 4, 5, 4, 4, 4, 5, 4, 1, 0, 0])
print("Data = ",data)`

Data = [4 4 4 4 4 8 7 4 4 2 6 3 5 8 6 6 4 3 1 4 4 9 2 7
4 7 6 5 3 4 5 4 4 5 4 10 0]

In [15]: `data.sort()
print(data)`

[0 1 2 2 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 5
5 5 5 6 6 6 6 7 7 7 8 8 8 9 10]

In [16]: `q1 = np.percentile(data,25)
q3 = np.percentile(data,75)
iqr = q3-q1
print(q1,q3,iqr)`

4.0 6.0 2.0

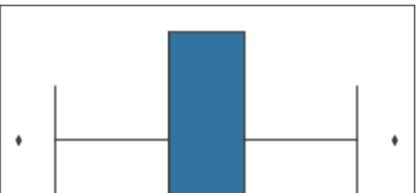
In [17]: `lcutoff = q1 - 1.5*iqr
ucutoff = q3 + 1.5*iqr
lcutoff,ucutoff`

Out[17]: (1.0, 9.0)

In [18]: `import seaborn as sns`

In [19]: `sns.boxplot(x=data)`

Out[19]: <matplotlib.axes._subplots.AxesSubplot at 0x7fb43f52aa58>





Multivariate Anomaly Detection

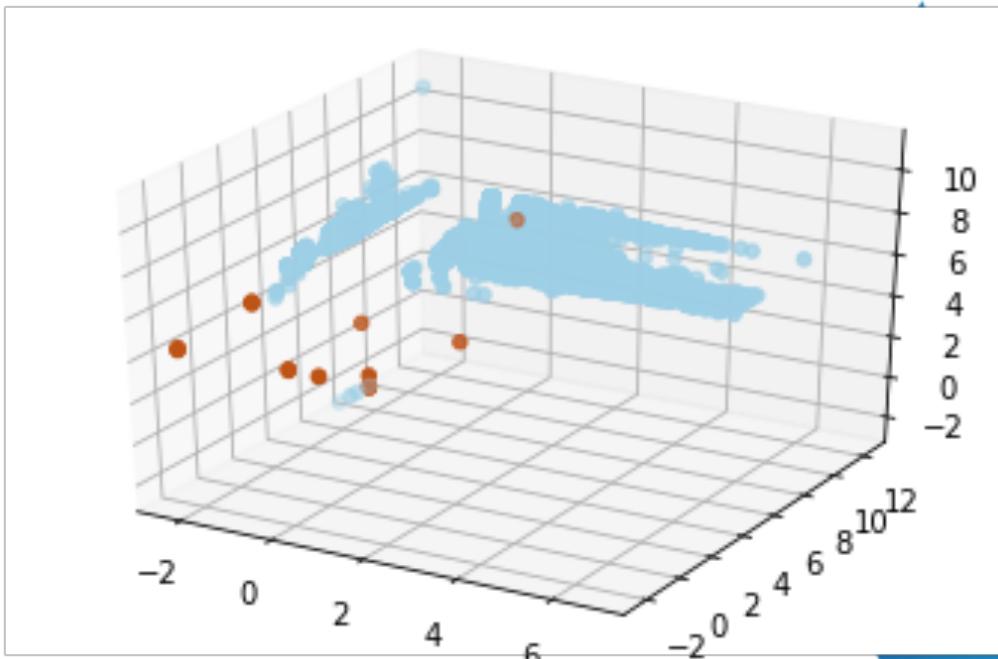




DB-Scan

	0	1	2	label
0	-3.351119	-1.325486	-4.510206	0
1	-6.762503	-1.837115	5.842649	0
2	-6.143649	6.226966	1.211651	0
3	-7.909054	4.304185	4.259787	0
4	-8.206218	2.532668	6.720400	0
...
1595	-4.761516	7.747586	7.979358	1
1596	-1.714259	-0.645442	-6.087254	1
1597	-2.198676	0.934126	-1.519782	1
1598	-2.424503	-4.967541	-8.313585	1
1599	3.550568	-5.749677	6.821268	1

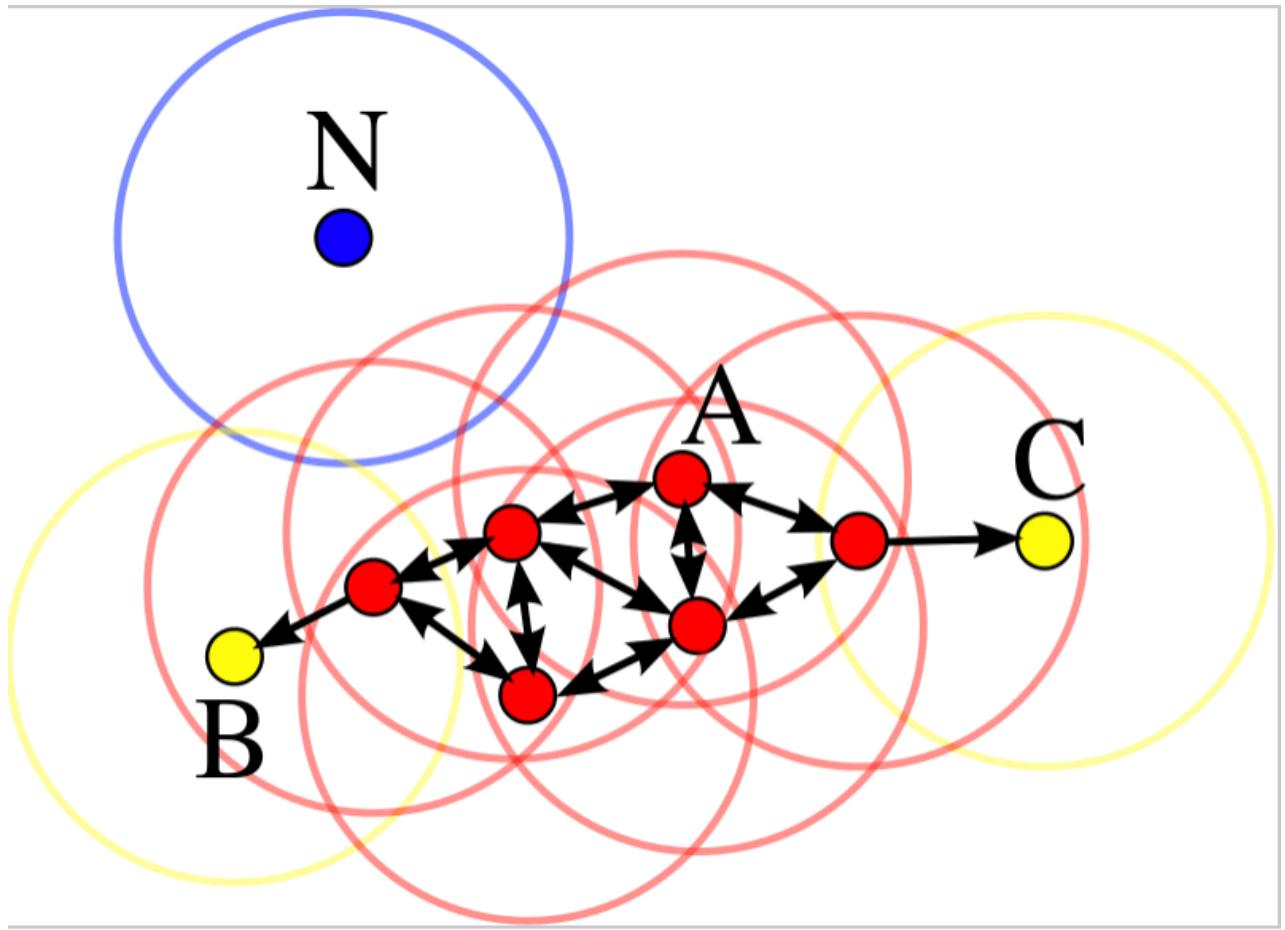
1600 rows × 4 columns



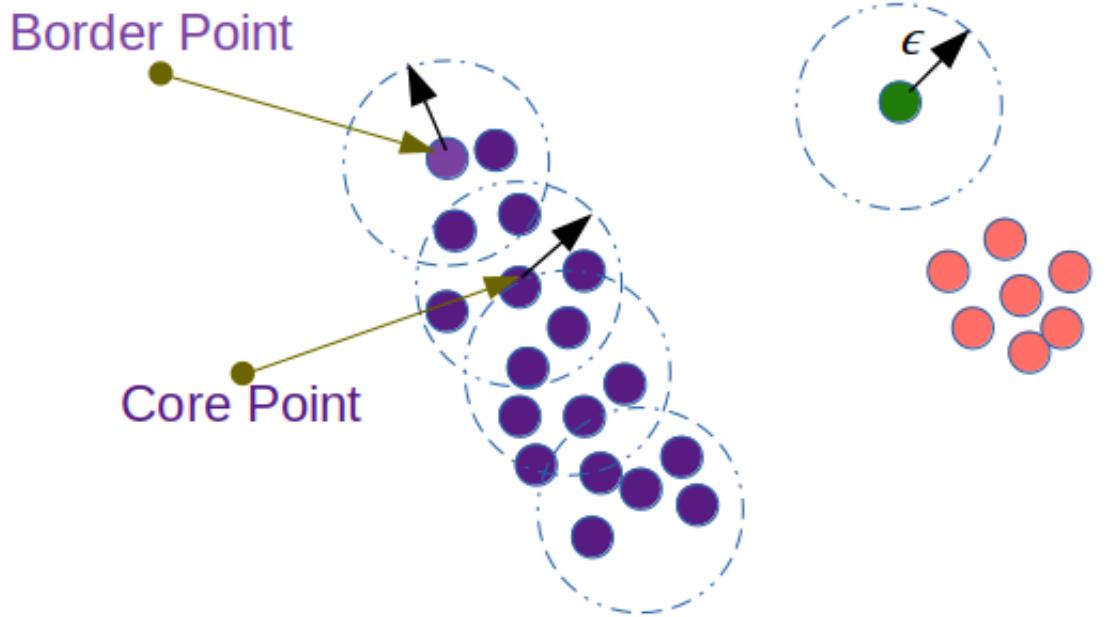


Anomaly Detection with DB-Scan





DB-Scan Explain

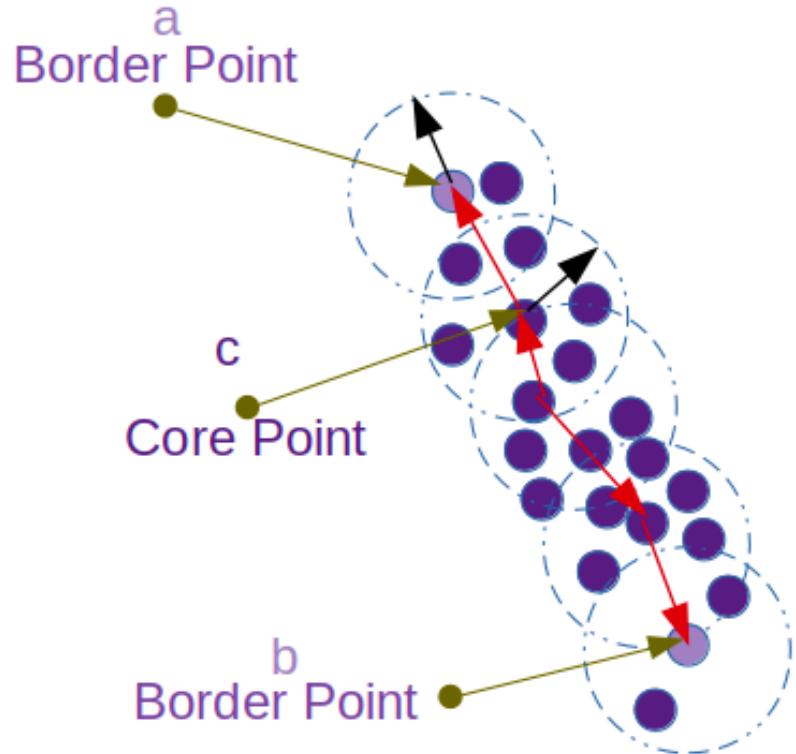


$$N_{Eps}(p) = \{q \in D \text{ such that } dist(p, q) \leq \epsilon\}$$

$\epsilon = 1$ unit, MinPts = 7



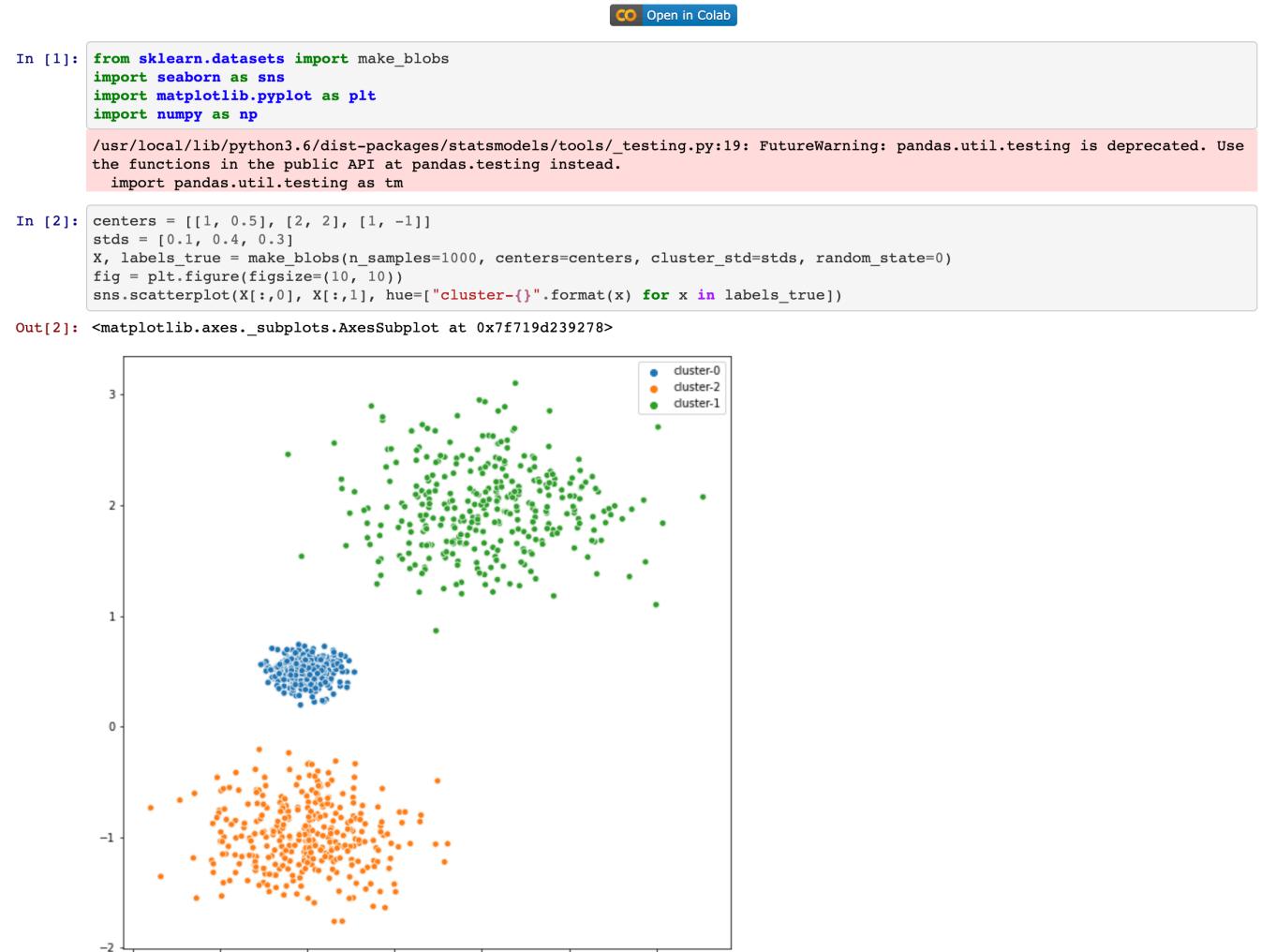
DB-Scan Explain



a, b are Density Reachable from a core point c.

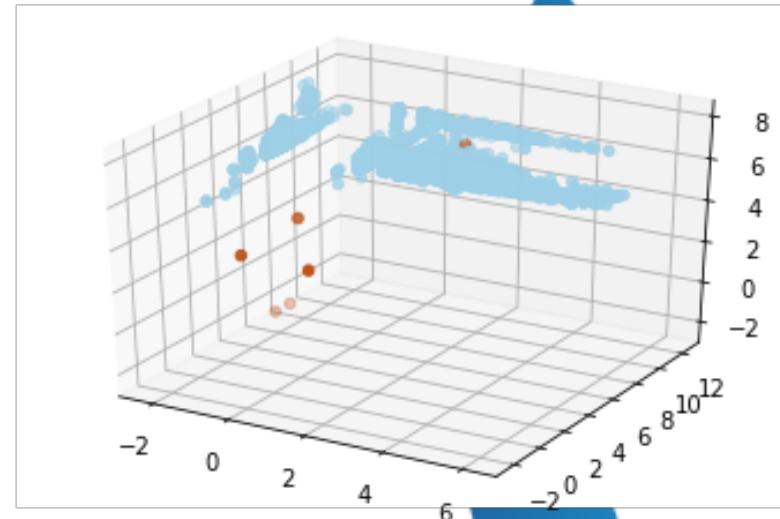
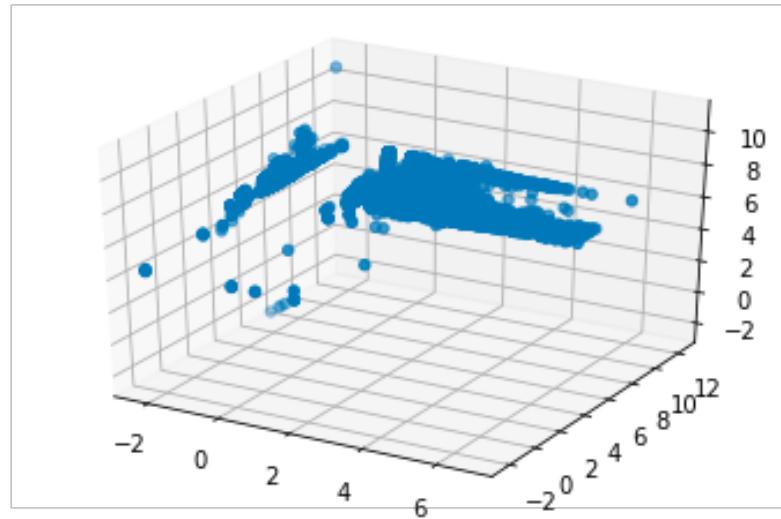
a, b are called Density Connected points.

DB-Scan Example



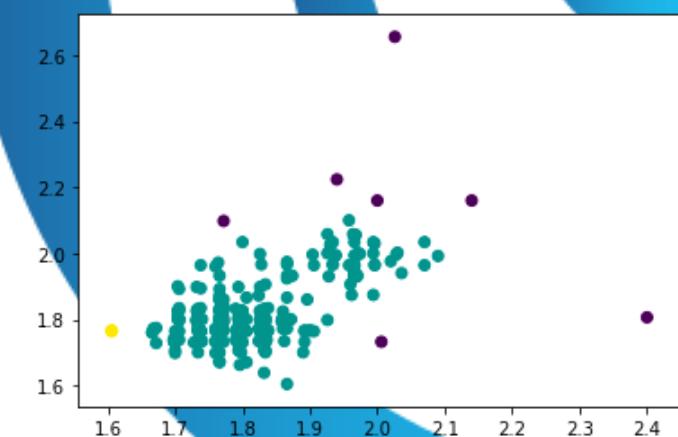
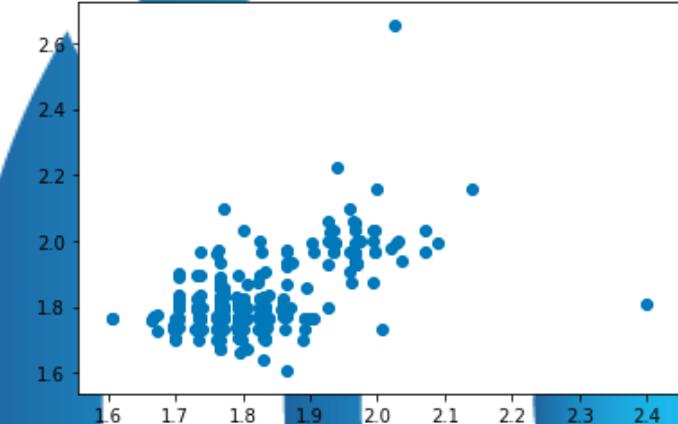
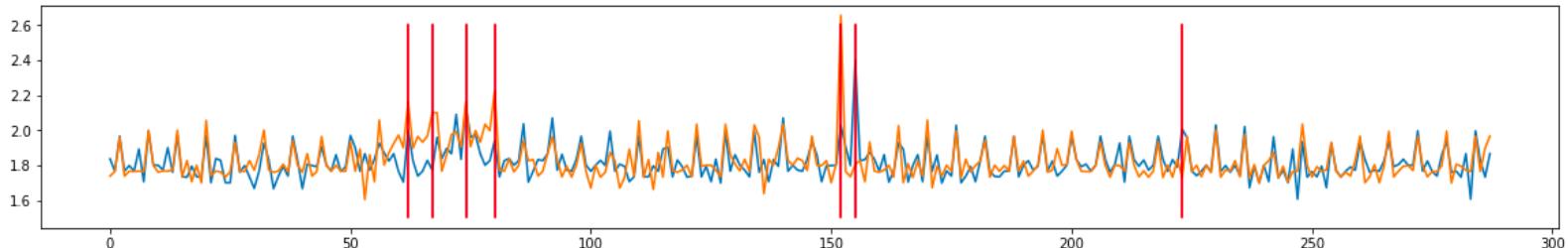
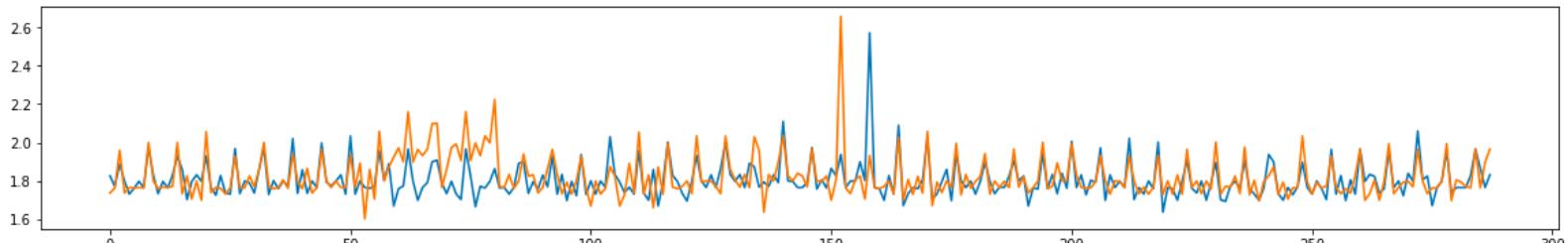


Anomaly with DB-Scan Example





DB-Scan Example

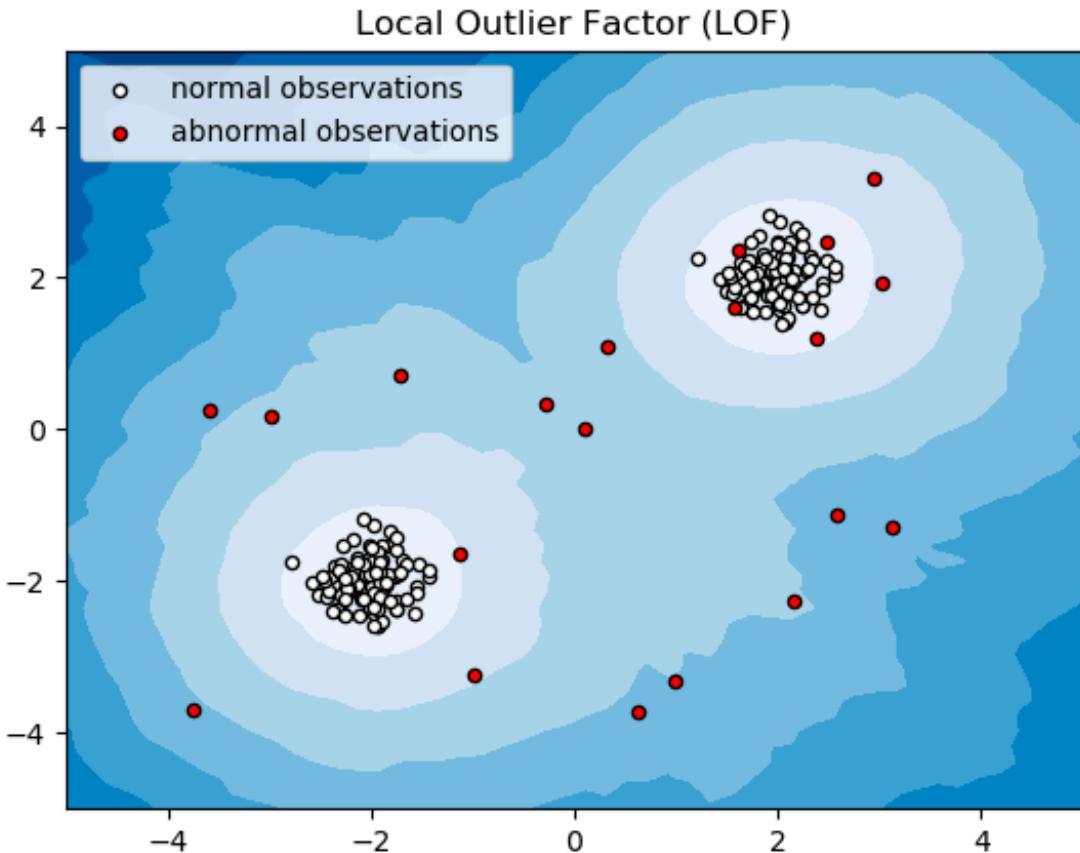




Anomaly Detection with Local Outlier Factor



Local Outlier Factor (LOF)



LOF Steps

- Reachability distance from o' to o :

$$\text{reachdist}_k(o \leftarrow o') = \max\{\text{dist}_k(o), \text{dist}(o, o')\}$$

– where k is a user-specified parameter

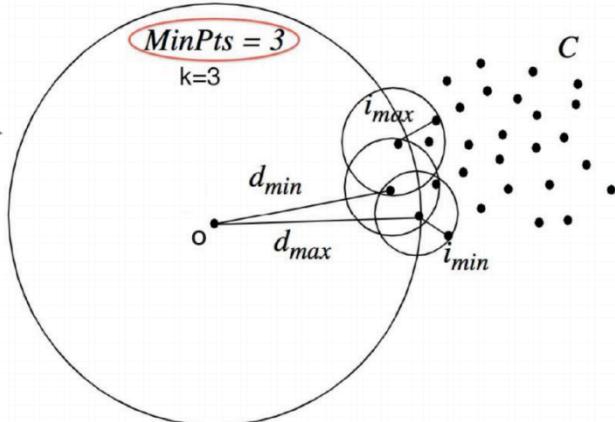
- Local reachability density of o :

$$\text{lrd}_k(o) = \frac{\|N_k(o)\|}{\sum_{o' \in N_k(o)} \text{reachdist}_k(o' \leftarrow o)}$$

- LOF (Local outlier factor) of an object o is the average of the ratio of local reachability of o and those of o 's k -nearest neighbors

$$\text{LOF}_k(o) = \frac{\sum_{o' \in N_k(o)} \frac{\text{lrd}_k(o')}{\text{lrd}_k(o)}}{\|N_k(o)\|} = \sum_{o' \in N_k(o)} \text{lrd}_k(o') \cdot \sum_{o' \in N_k(o)} \text{reachdist}_k(o' \leftarrow o)$$

- The lower the local reachability density of o , and the higher the local reachability density of the kNN of o , the higher LOF
- This captures a local outlier whose local density is relatively low comparing to the local densities of its kNN



LOF Calculation

Usaging Manhattan distance with k=2

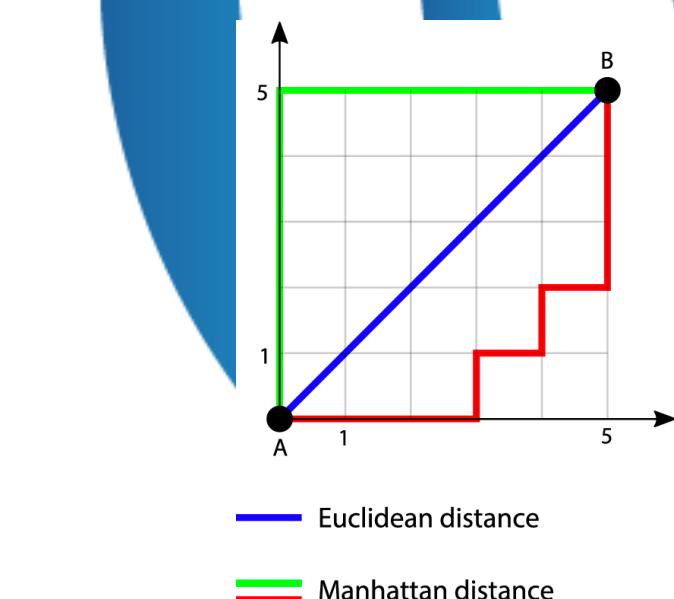
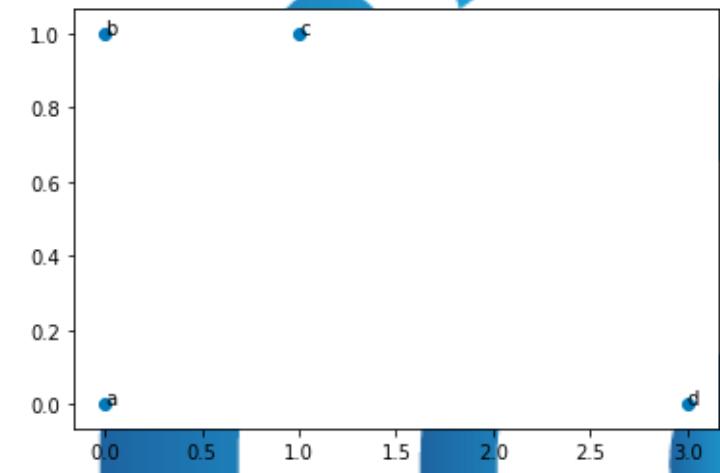
$\text{dist}(a,b) = 1$
 $\text{dist}(a,c) = 2$
 $\text{dist}(a,d) = 3$
 $\text{dist}(b,c) = 1$
 $\text{dist}(b,d) = 4$
 $\text{dist}(c,d) = 3$

$\text{dist}(a) = 2$ longest distance is a-c
 $\text{dist}(b) = 1$ longest distance is b-a , b-c
 $\text{dist}(c) = 2$ longest distance is c-a
 $\text{dist}(d) = 3$ longest distance is d-a , d-c

Reachability Density

Ex. $\text{rd}(b <- a) = \max(\text{dist}(b), \text{dist}(b,a))$

$\text{rd}(b <- a) = \max(1,1) = 1$
 $\text{rd}(a <- b) = \max(2,1) = 2$
 $\text{rd}(c <- a) = \max(2,2) = 2$
 $\text{rd}(d <- a) = \max(3,3) = 3$
 $\text{rd}(c <- b) = \max(2,1) = 2$
 $\text{rd}(d <- b) = \max(3,4) = 4$
 $\text{rd}(c <- d) = \max(3,3) = 3$





LOF Example

master ▾ AnomalyDetection / Anomaly_LOF_Example.ipynb Go to file ...

krmonline Created using Colaboratory Latest commit e292b63 now History

1 contributor

350 lines (350 sloc) | 32.1 KB

<> Open in Colab Raw Blame

```
In [ ]: import numpy as np
import h5py
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.neighbors import LocalOutlierFactor

In [ ]: clf = LocalOutlierFactor(n_neighbors=2, contamination=0.4)
(X,Y) = np.asarray([[0,0],[0,1],[1,1],[3,0]]).T
y_pred = clf.fit_predict(np.asarray([X,Y]).T)
clf.negative_outlier_factor_

Out[ ]: array([-0.9267767 , -1.17157288, -0.9267767 , -2.16885037])

In [ ]: plt.scatter(X,Y)
plt.text(0,0,'a')
plt.text(0,1,'b')
plt.text(1,1,'c')
plt.text(3,0,'d')

Out[ ]: Text(3, 0, 'd')
```



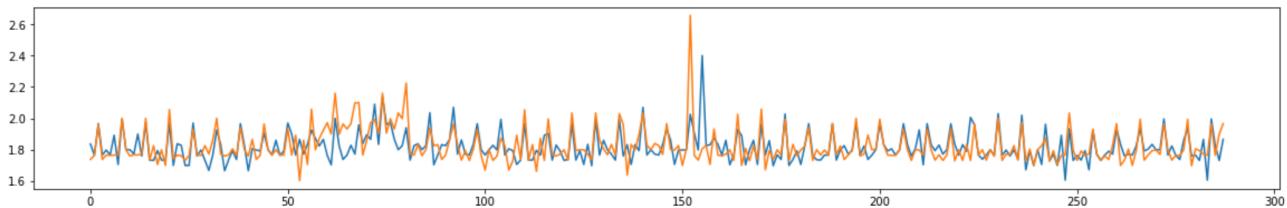


LOF Example #2

```
In [ ]: X = cpu2[4]
Y = cpu2[5]
clf = LocalOutlierFactor(n_neighbors=20, contamination=0.03)
y_pred = clf.fit_predict(np.asarray([X,Y]).T)
```

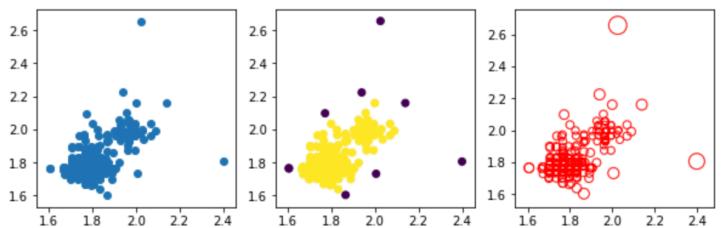
```
In [ ]: plt.figure(figsize=(20, 3))
plt.plot(X)
plt.plot(Y)
```

```
Out[ ]: <matplotlib.lines.Line2D at 0x7fbcd6403aba8>
```



```
In [ ]: plt.figure(figsize=(10, 3))
plt.subplot(131).scatter(X,Y)
plt.subplot(132).scatter(X,Y,c=y_pred)
plt.subplot(133).scatter(X,Y,s=30*np.abs(clf.negative_outlier_factor_),facecolors='none',edgecolors='r')
```

```
Out[ ]: <matplotlib.collections.PathCollection at 0x7fbcd63f46f98>
```

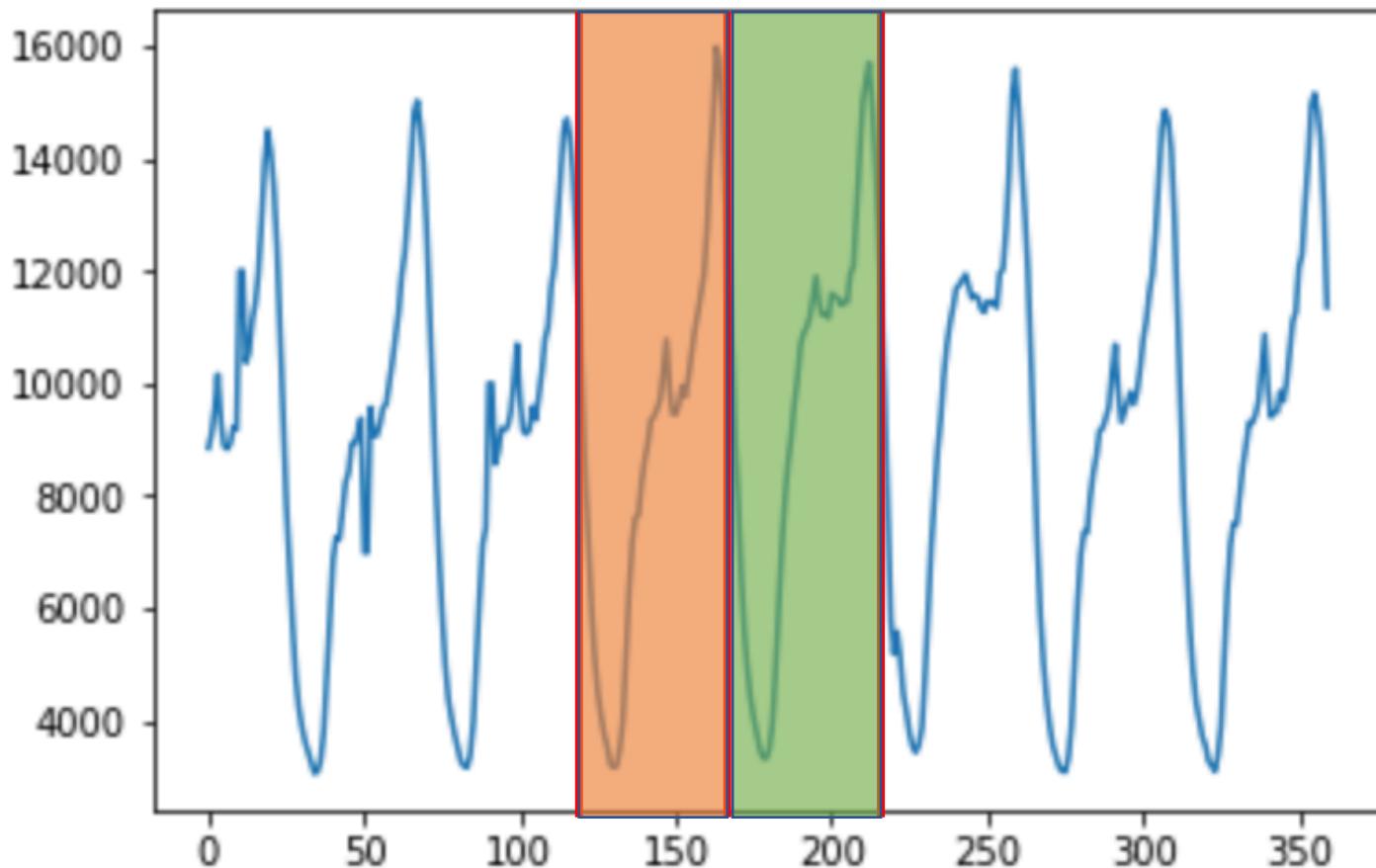


```
In [ ]: plt.figure(figsize=(20, 3))
plt.plot(X)
plt.plot(Y)
for i in range(len(y_pred)):
    if y_pred[i] == -1:
        plt.plot([i,i],[1.5,2.6], 'r-')
```



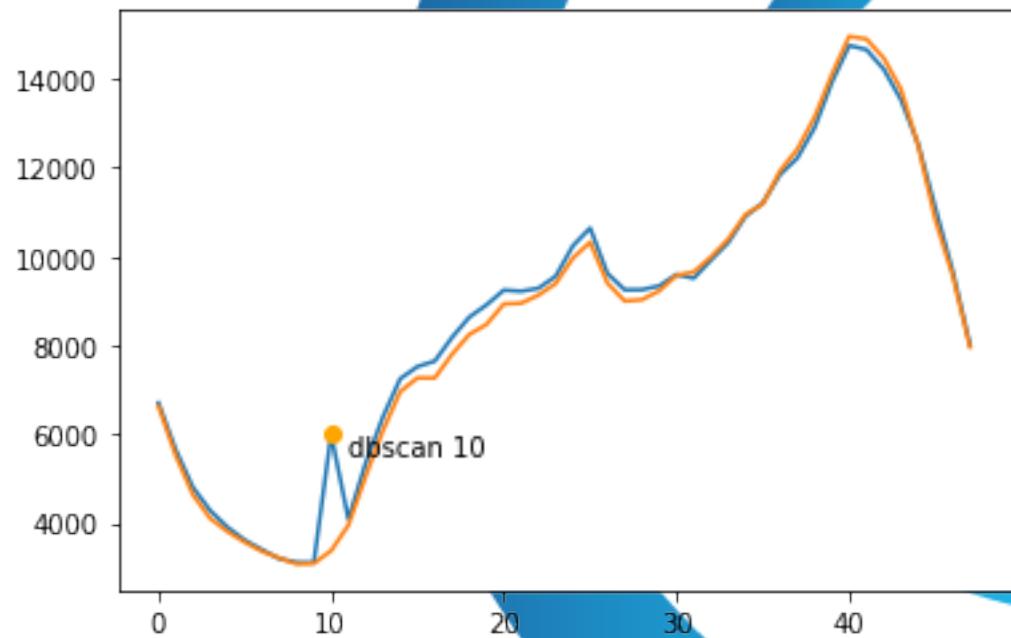
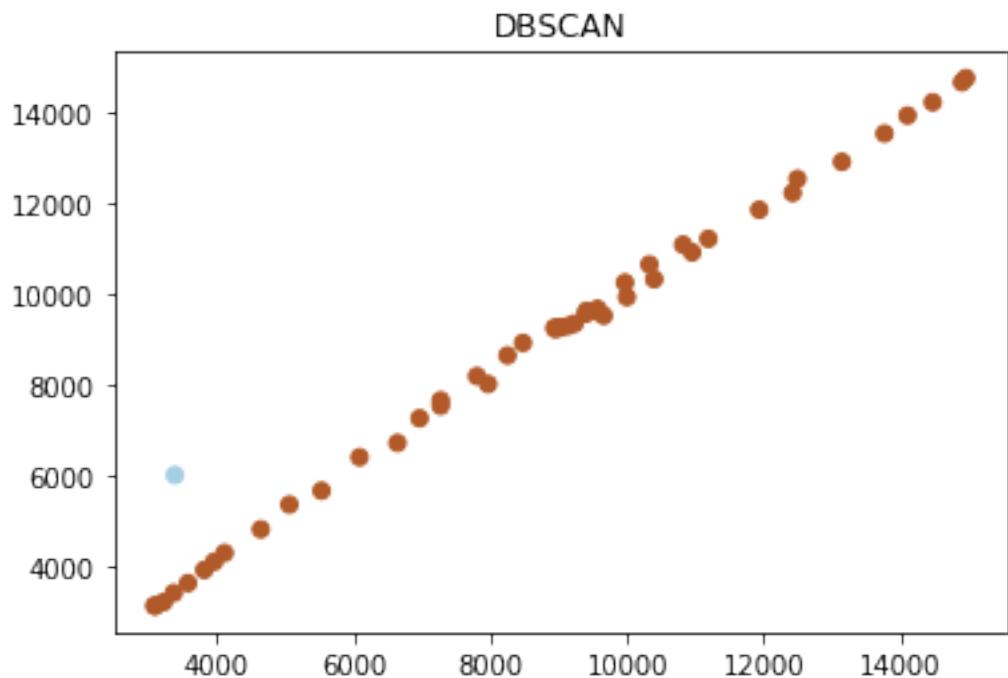
Time series

Converse 1 dim to 2 dim



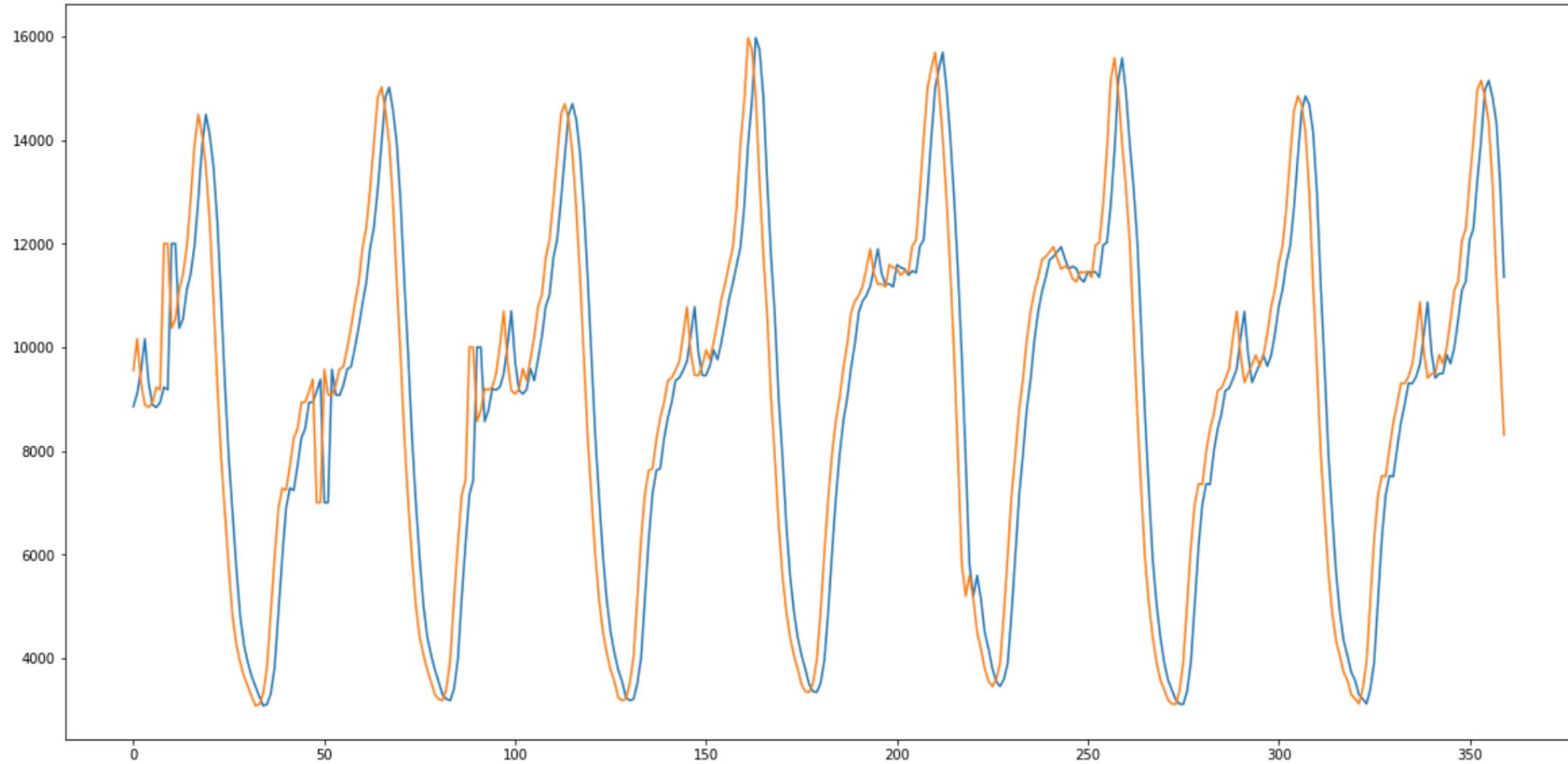
	dataset1	dataset2
0	6636.93	6708.39
1	5525.74	5654.90
2	4636.56	4803.93
3	4106.13	4282.68
4	3809.40	3908.50
5	3571.48	3616.86
6	3362.99	3401.56
7	3215.41	3199.62
8	3079.79	3118.59
9	3093.17	3113.45
10	3380.14	3467.37
11	3954.25	4088.98
12	5054.78	5349.37
13	6082.58	6393.57
14	6960.80	7251.71
15	7269.43	7525.05

Time series

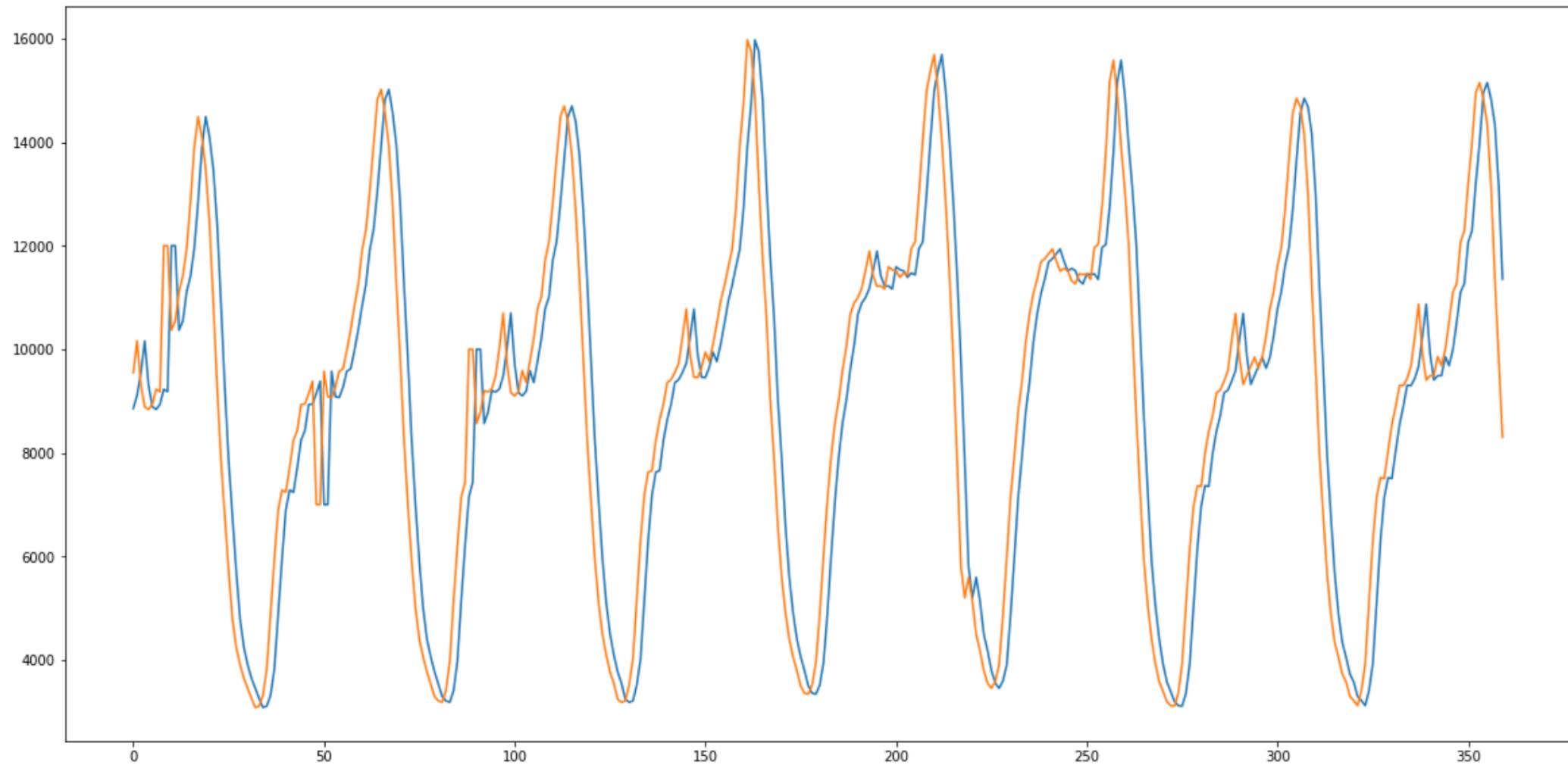




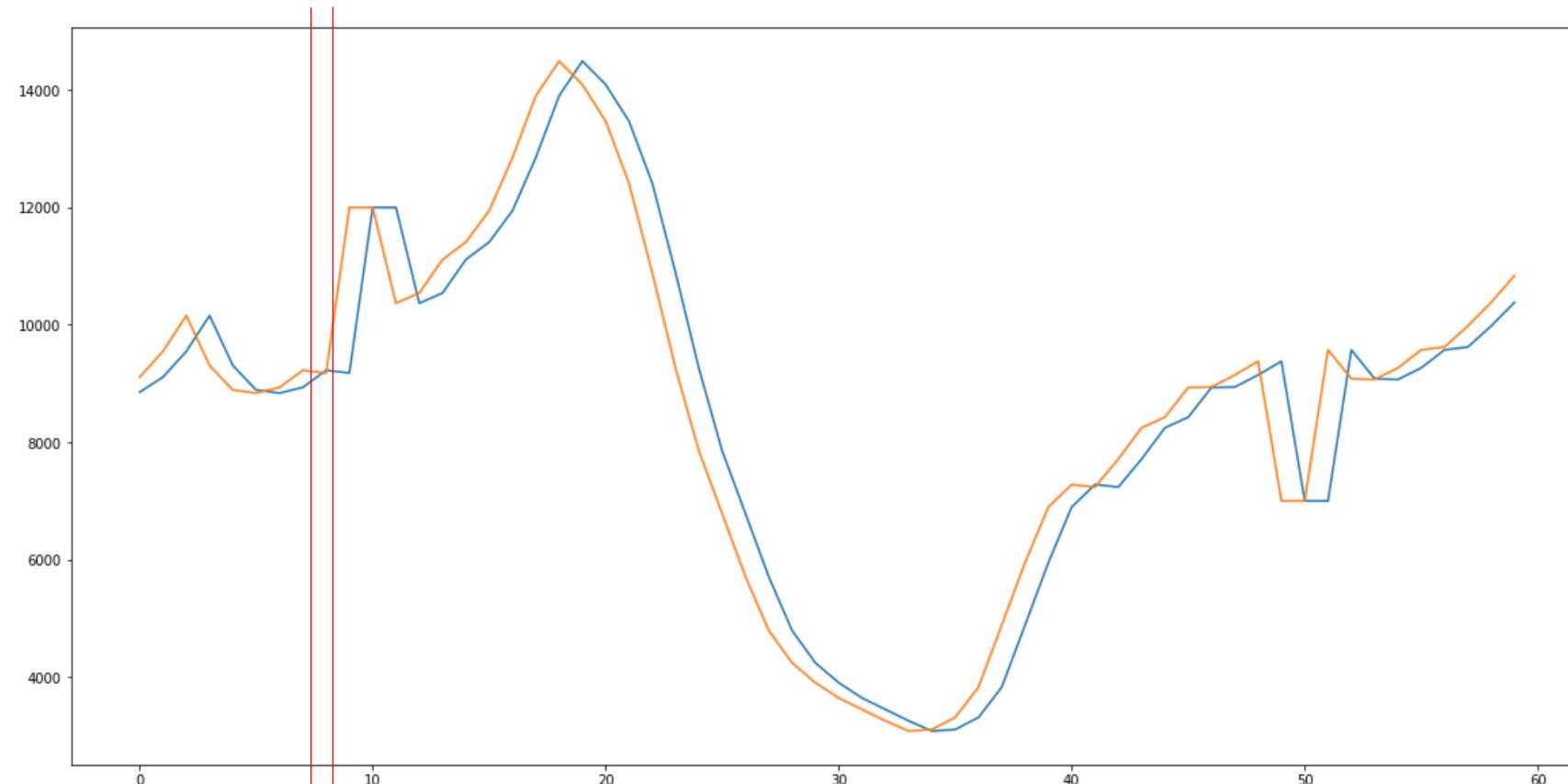
Time Shift Detection



Create new graph with shift to 1 step

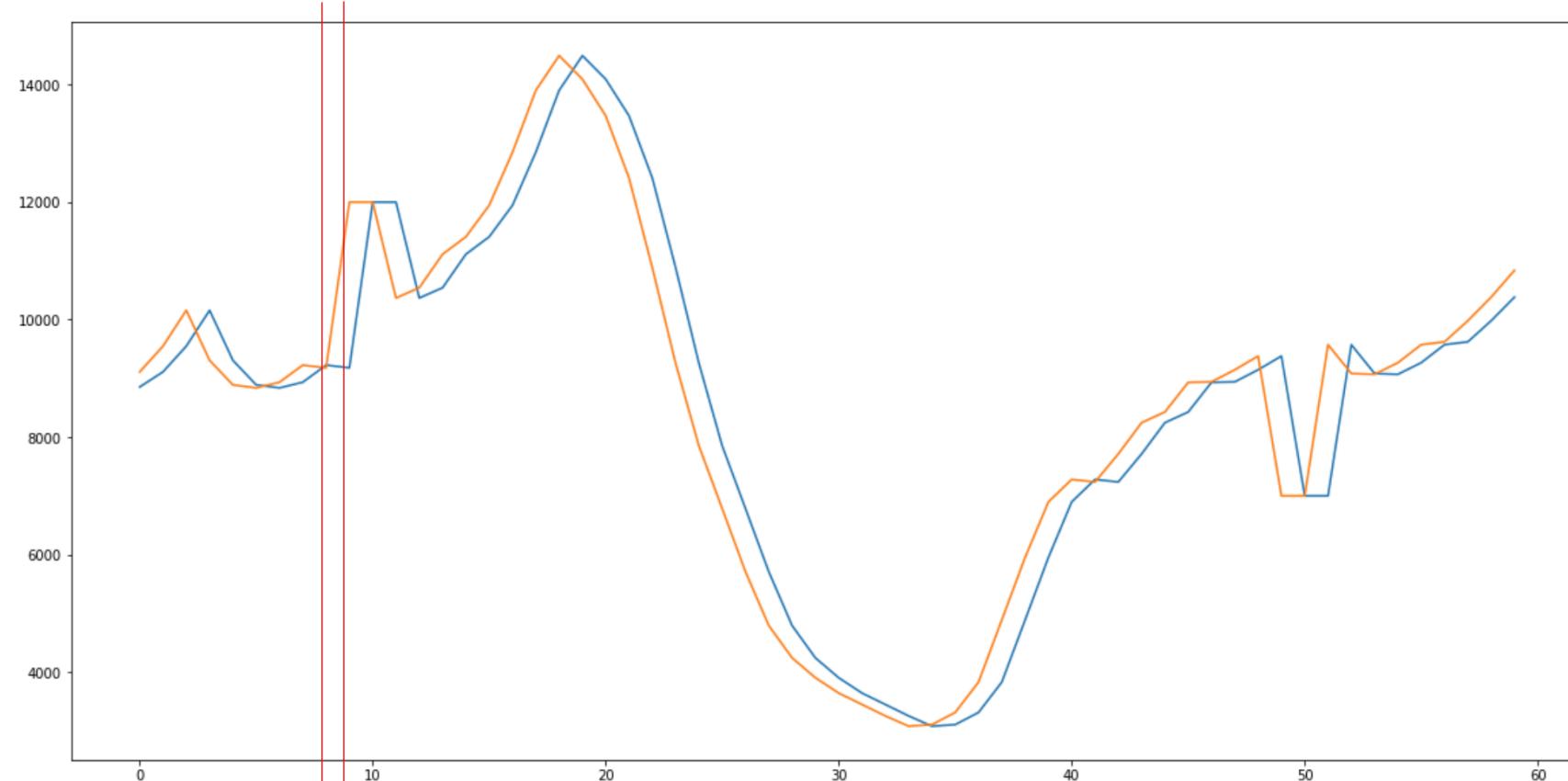


Zoom your graph and shift first 3 step



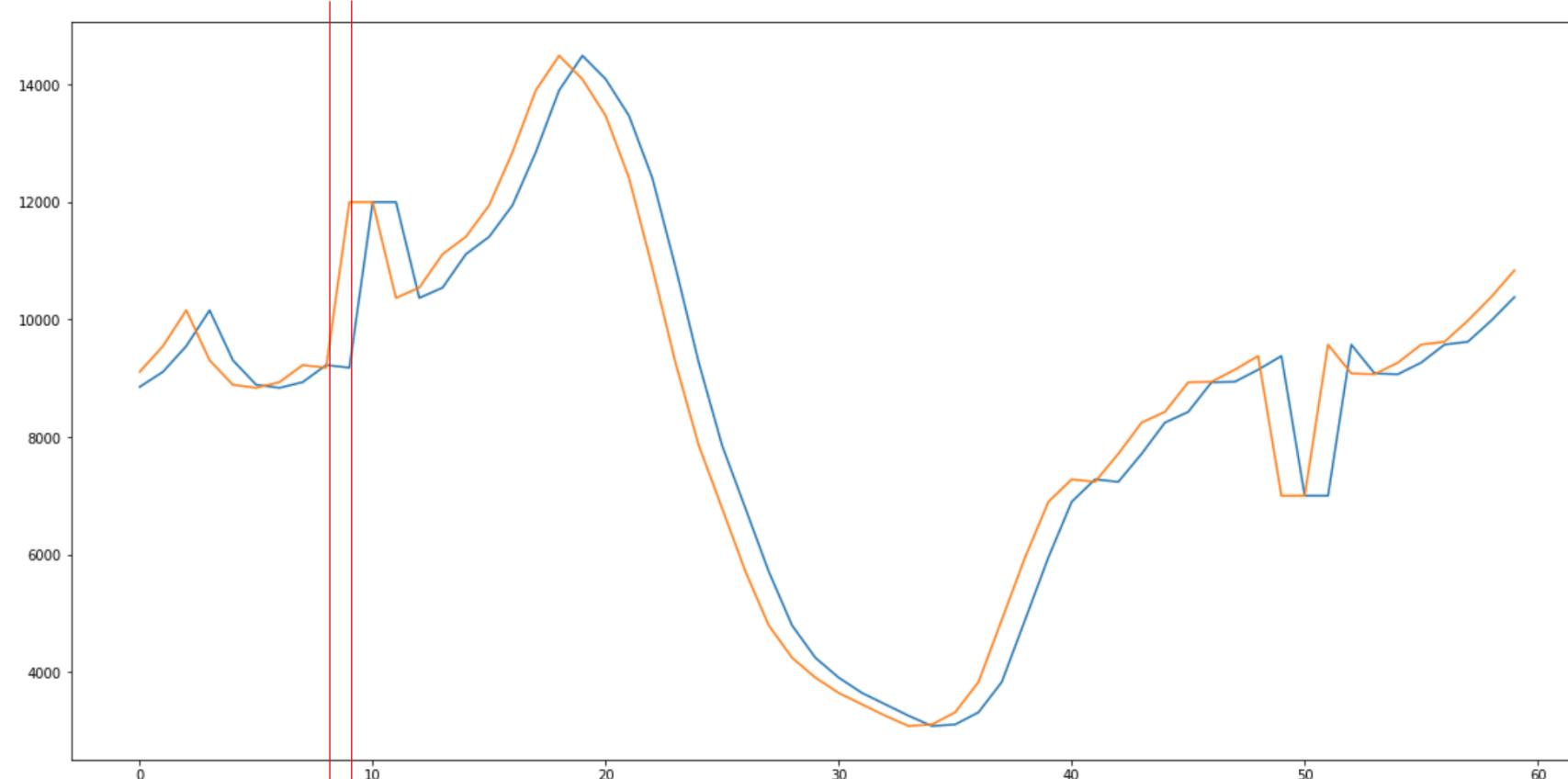
0 . 995416134063

Zoom your graph and shift first 3 step



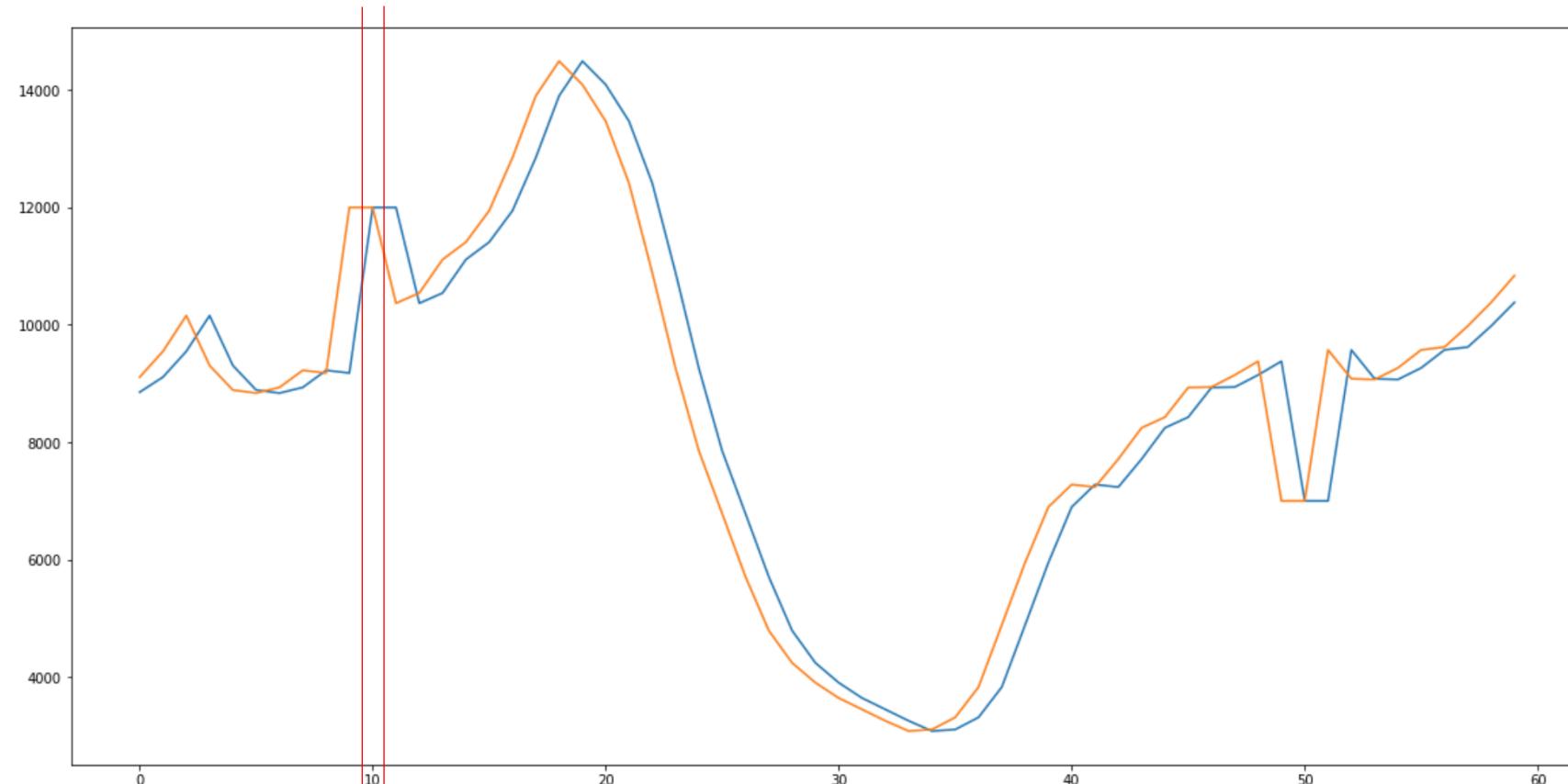
0 . 995416134063

Zoom your graph and shift first 3 step



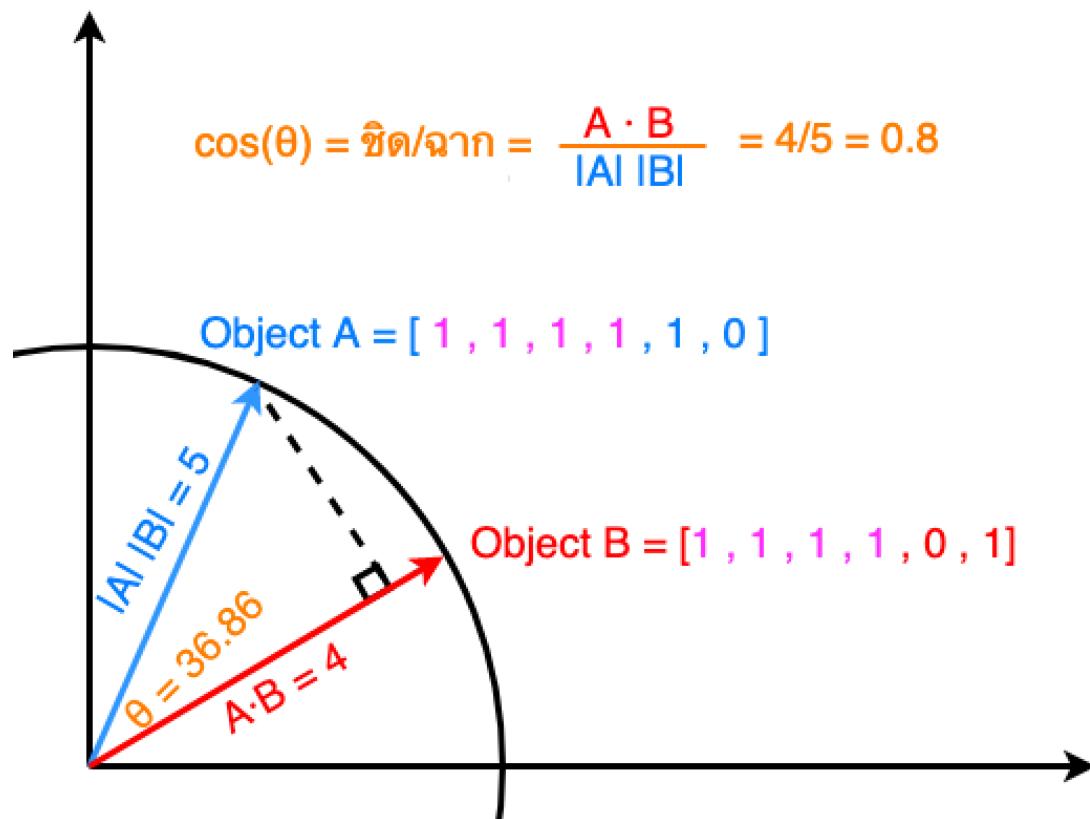
0 . 995416134063

Zoom your graph and shift first 3 step

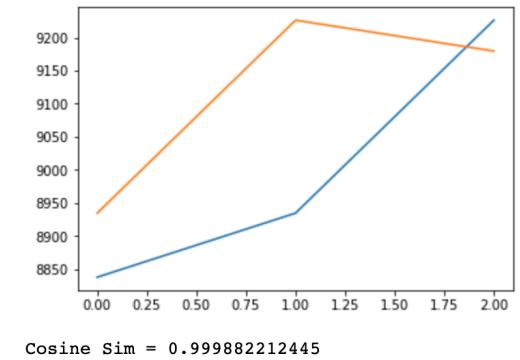
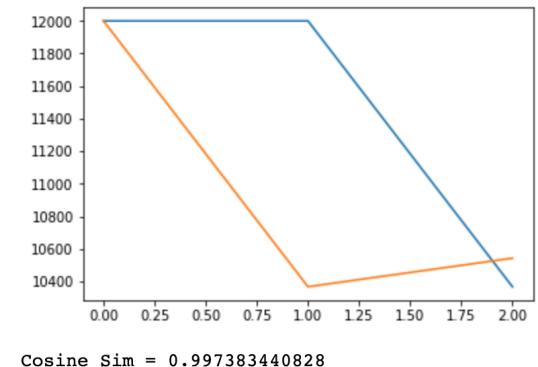
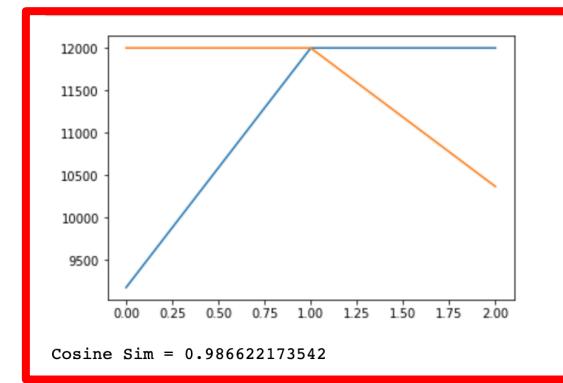
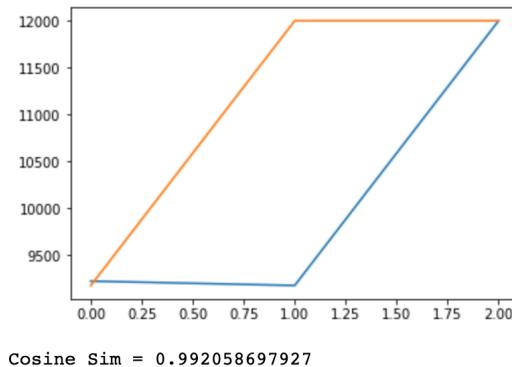
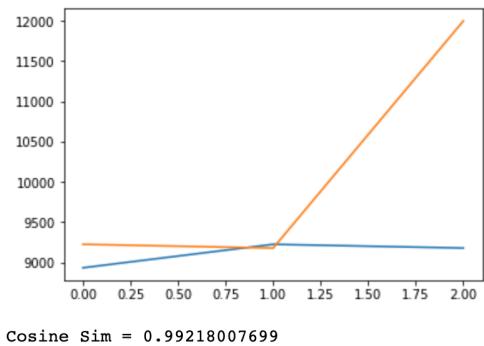


0 . 995416134063

Cosine Similarity



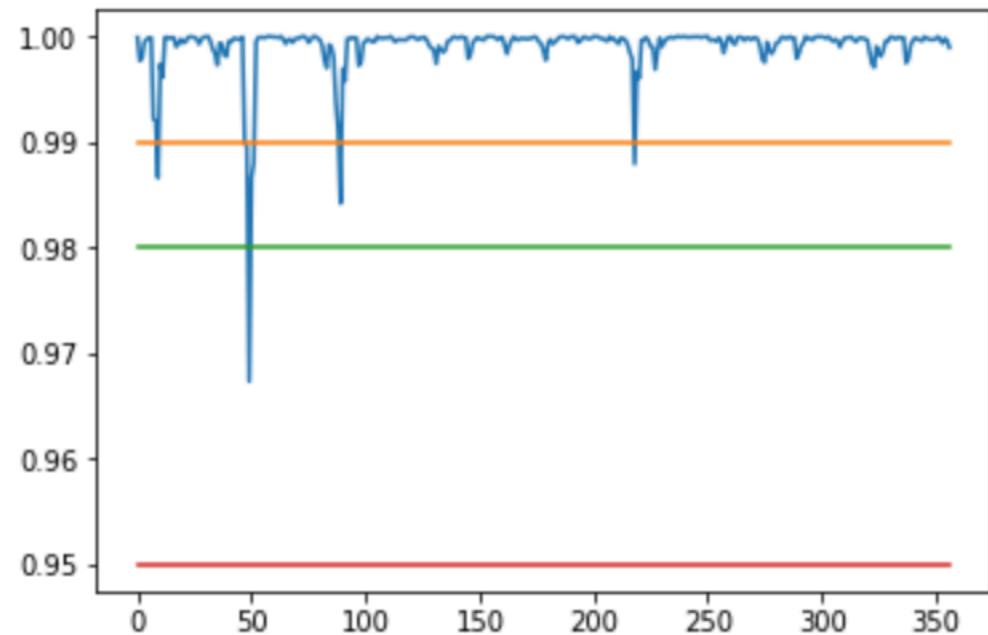
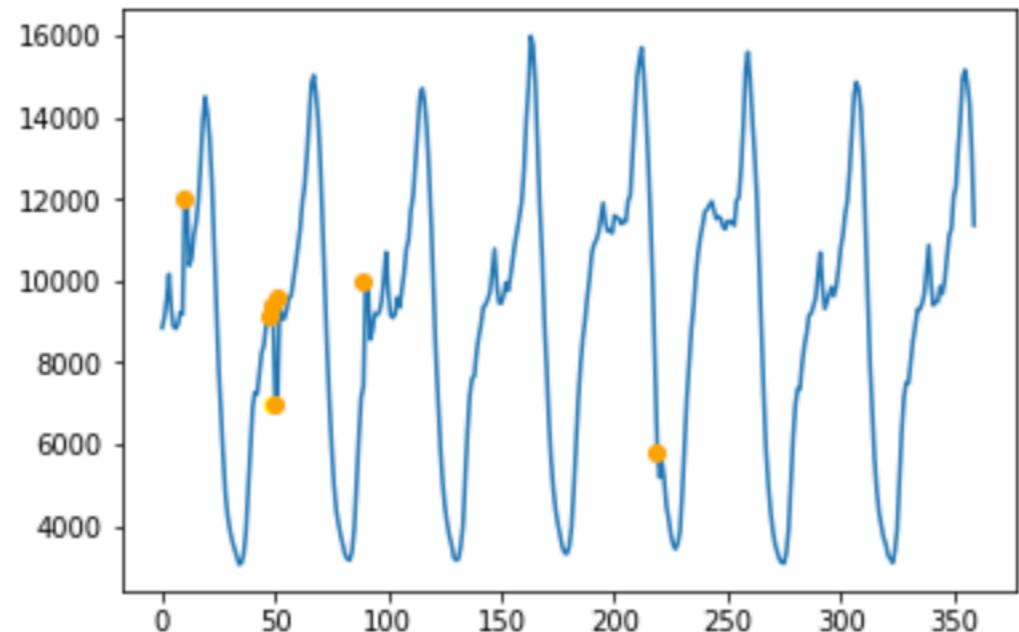
Rotate graph next 3 step and compare with cosine similarity



Remark when value lower than threshold



Anomaly Detection and Cosine Values





Time Shift Detection Example

```
In [10]: a = np.array([1,2,3,4,5,6,7,7,7,7])
b = np.array([5,5,6,6,7,7,8,8,8,9])
```

```
print(a,b)
```

```
[1 2 3 4 5 6 7 7 7 7] [5 5 6 6 7 7 8 8 8 9]
```

```
In [11]: from sklearn.preprocessing import StandardScaler
#inverse_transform
sa = StandardScaler().fit(a.reshape(-1,1))
sb = StandardScaler().fit(b.reshape(-1,1))
a_Transform = sa.transform(a.reshape(-1,1))
b_Transform = sb.transform(b.reshape(-1,1))
print(a_Transform)
#print(b_Transform)
```

```
[[ -1.80085268]
```

```
[ -1.33909558]
```

```
[ -0.87733848]
```

```
[ -0.41558139]
```

```
[  0.04617571]
```

```
[  0.50793281]
```

```
[  0.9696899 ]
```

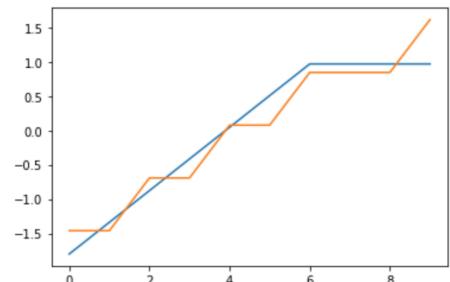
```
[  0.9696899 ]
```

```
[  0.9696899 ]
```

```
[  0.9696899 ]
```

```
In [12]: #print(a_Transform)
plt.plot(a_Transform.reshape(1,-1)[0])
plt.plot(b_Transform.reshape(1,-1)[0])
```

```
Out[12]: [<matplotlib.lines.Line2D at 0x7fbcb56318b70>]
```



```
In [13]: #print(a_Transform)
```