Anomaly Detection

Yoni Mayzel

Adapted from Morris Alper, Yoni Krichevsky

Agenda

< itc >

- Motivation
- Intro to anomaly detection: types, challenges
- Local Outlier Factor (LOF)
- Isolation Forest
- Anomaly detection with autoencoders
- Anomaly detection metrics

Motivation

Anomaly Detection – Use cases

Detecting **credit card fraud**

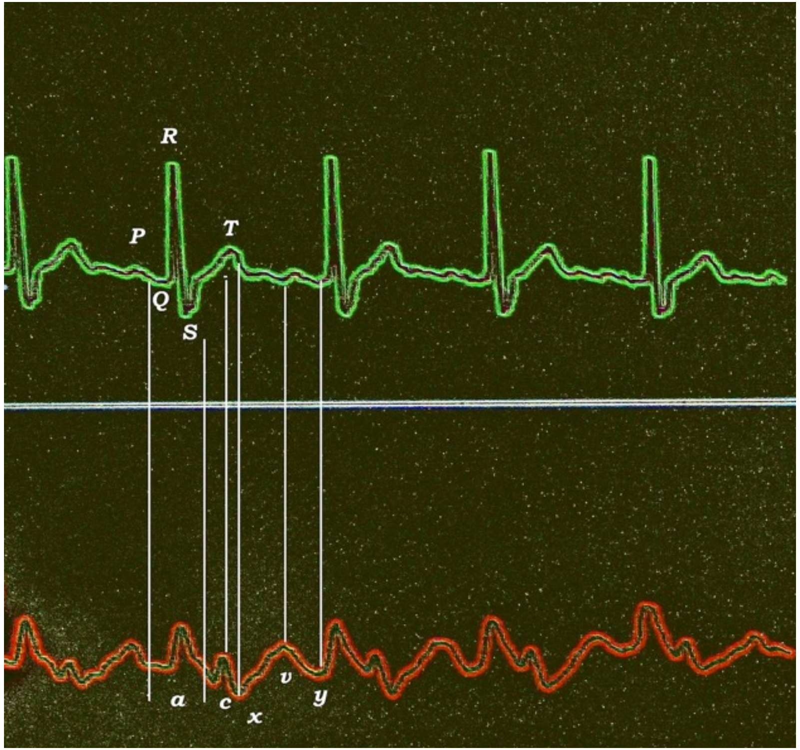
A cardholder makes tens of thousands of transactions, but there is a chance that one or two are fraudulent.

How can we **automatically flag suspicious** transactions?

Anomaly Detection – Use cases

Patients in a hospital are connected to **health monitoring systems.**

Usually all is well, but **we must be alerted if there are unusual signs.**

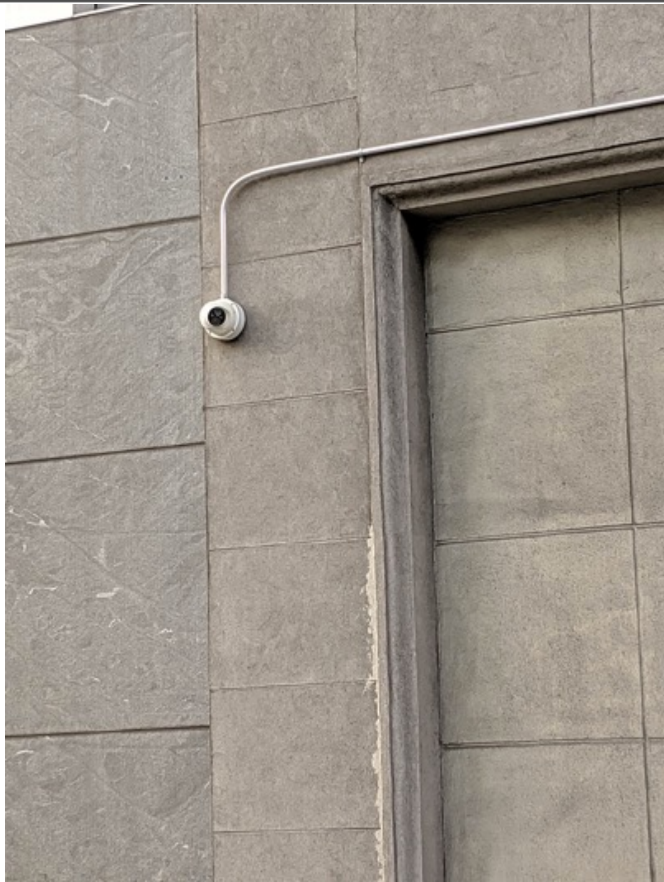


Anomaly Detection – Use cases

A **surveillance system** is monitoring a private home.

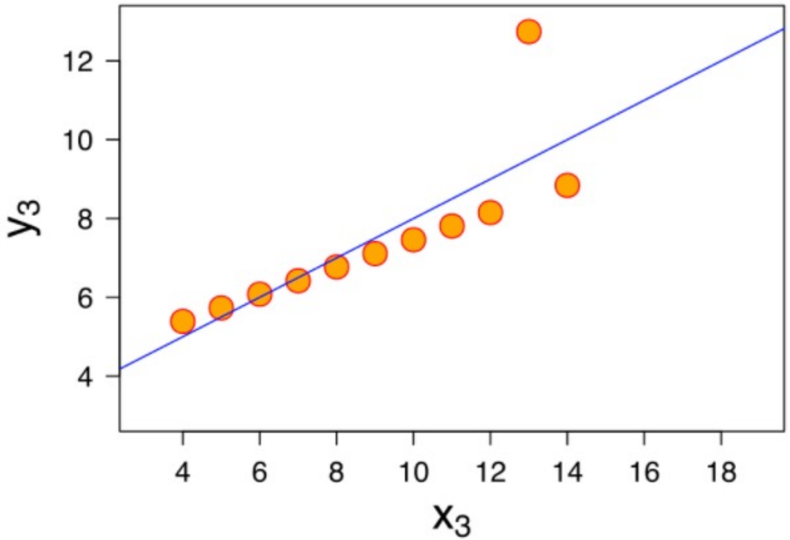
Usually there are no events, but we must raise an alarm if there is a likely intruder.

How can we **detect intrusions?**



Anomaly Detection – Motivation

Common idea: detecting **outliers**, data points that do not pattern with the dataset as a whole



Anomaly Detection vs Classification

Q: Why can’t we just use the existing methods we have learned for supervised binary classification? (0=inlier, 1=outlier)

A: A few challenges:

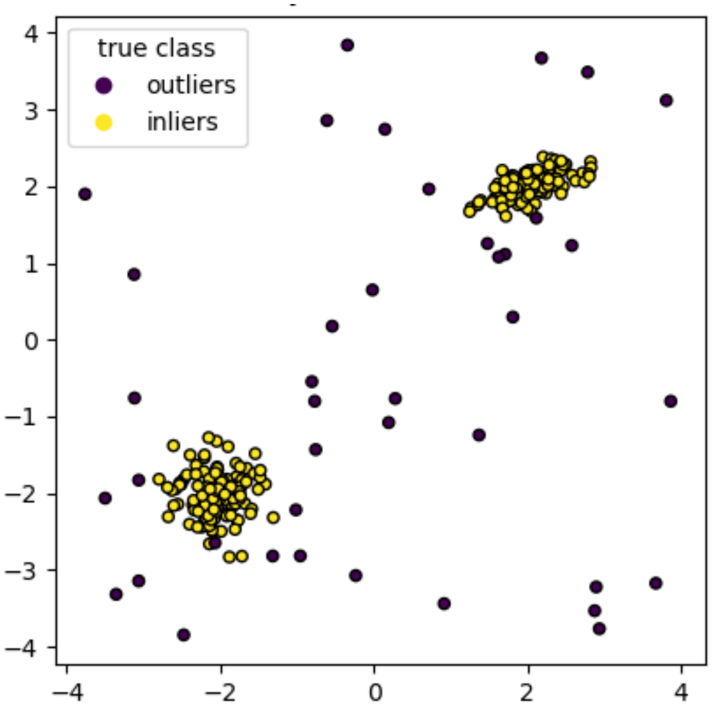
- Very unbalanced - outliers are usually rare, sometimes very rare
- Often no labelled data (unsupervised)
- Outliers may not share features or form a cluster

Intro to anomaly detection

Synonyms

Outliers = Anomaly = exception

Opposite: inlier



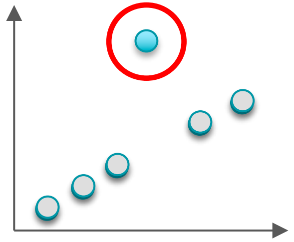
Types: Outlier vs Novelty

Outliers: deviant samples in existing dataset

Novelties: new samples that do not fit the distribution of the existing dataset

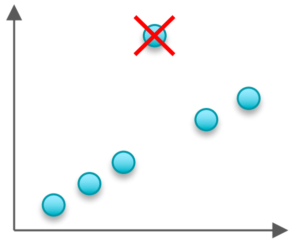
For this talk we will focus on **outlier detection**, but some of the methods can be used for **novelty detection** as well.

Types: AD vs Noise removal



Didn’t we learn already about removing outliers?

Anomaly detection – we are interested in <u>finding</u> the anomalies



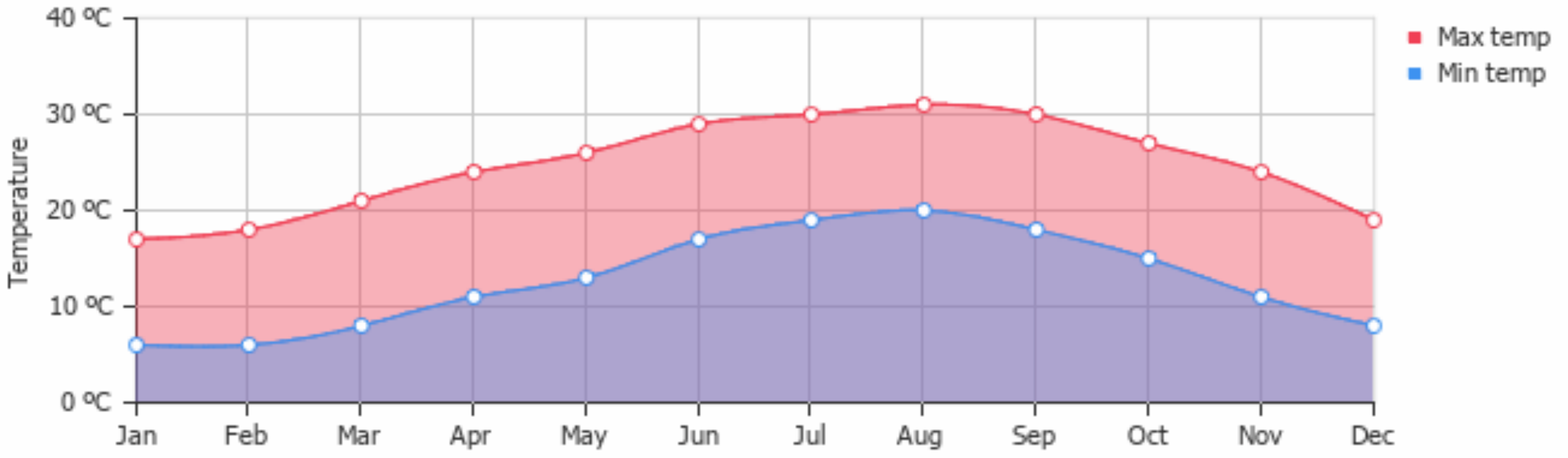
Noise removal (outlier removal) **–** we are interested <u>removing</u> the noise, stay with data without noise

Q: can we use what we learned about removing outliers for anomaly detection?

Types: Global vs contextual

Q: If the temperature in Tel Aviv is 28C in a specific day, it is an outlier?

A: That depends on the season.

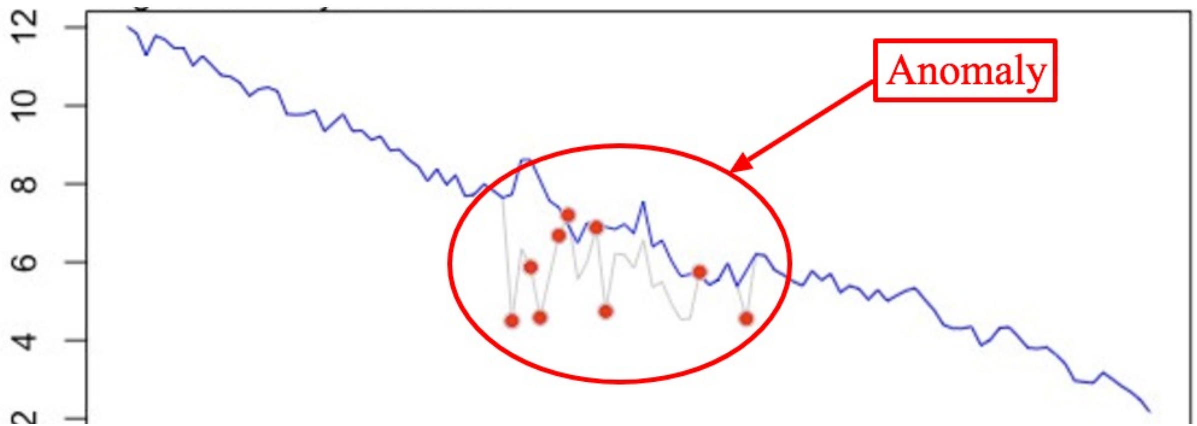


Average min and max temperatures in Tel Aviv, Israel Copyright © 2019 www.weather-and-climate.com

Types: Global vs contextual

A **global outlier** deviates from the rest of the dataset as a whole

A **contextual outlier** deviates based on a context (e.g. time)



Types: supervised vs unsupervised

Unsupervised anomaly detection (more common)

- Finds outliers without any labels given in advance

Supervised anomaly detection (less common)

- Trains a binary classifier on training data labelled as “normal” and “abnormal”

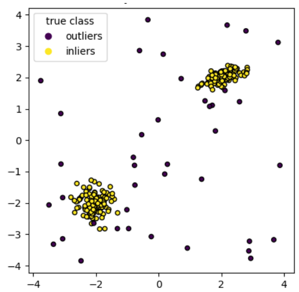
The methods we will cover are all **unsupervised**

Challenges

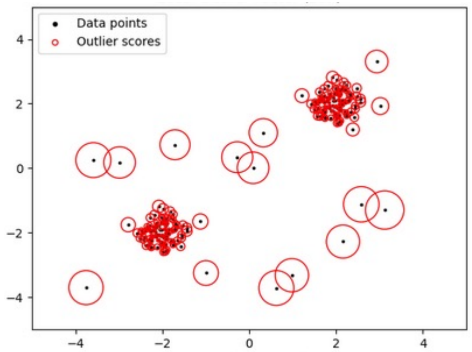
- In malicious activity, attackers will work to make it look normal
- Normal activity changes over time
- Domain specific
  - In some domains (medical), even a small change might be anomaly, in others (stock market), even large ones might not be
- Difficulty to differentiate between anomalies and noise
- Just like with classification, there is a tradeoff between FP & FN

Output of algorithms

- **Binary prediction**
  - outlier (1), inlier (0)
  - -1 (outlier), 1 (inlier)



- **Score / rank** – to what extent it's an outlier. Then we can take X most "unusual" observations

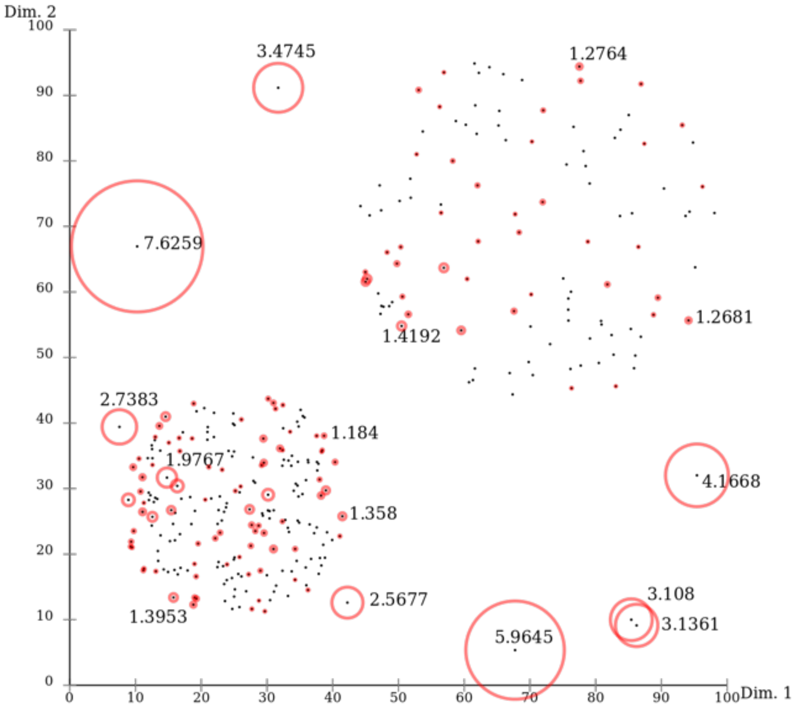


Local Outlier Factor (LOF) – Density Based Family

Local Outlier Factor (LOF) – Density Based

Local Outlier Factor (LOF) assumes outlier are located in locally low-density areas in feature space.

If point x is located in a lower density area compared to the density of its k-closest neighbors, then maybe x is an outlier



Local Outlier Factor (LOF) – Density Based

Each point in the dataset is assigned a score: **Local Outlier Factor (LOF)**

LOF < 1:

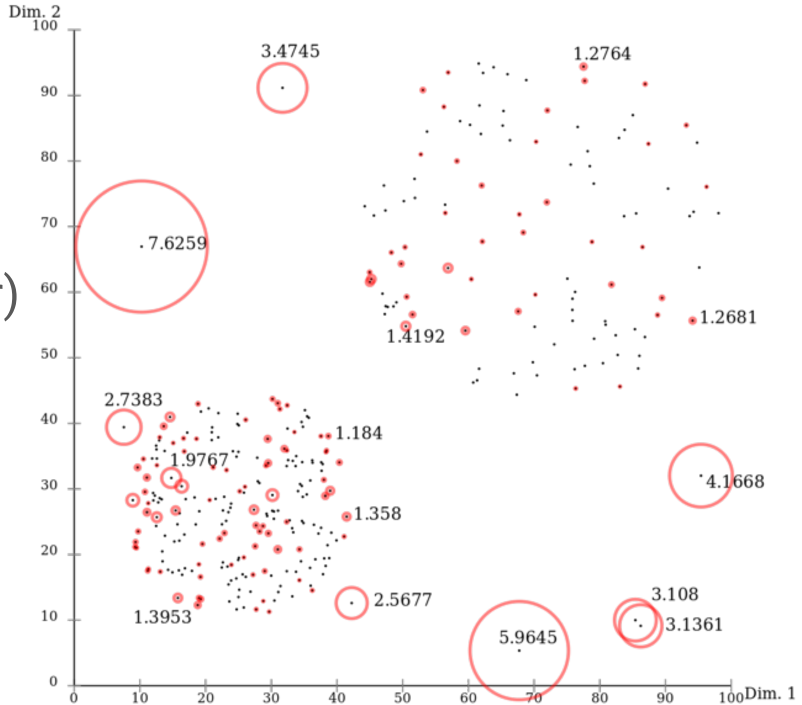
- high density compared to its neighbors (inlier)

LOF ≈ 1:

- similar density to its neighbors

LOF > 1:

- low density compared to its neighbors (outlier)



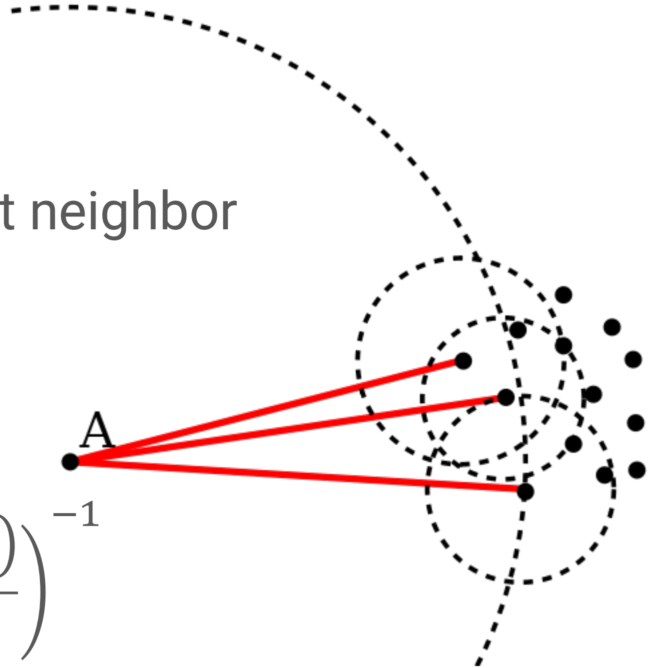
Local Outlier Factor (LOF) calculation

Definition of LOF$_k$:

$d(A, B)$: distance from sample A to B

$d_k(B)$: distance from sample B to its k-th nearest neighbor

$N_k(A)$: set of k nearest neighbors to A



The ***local reachability density*** of A:

$$lrd_k(A) = \left(\frac{\sum_{B \in N_k(A)} \max\big(d(A, B),\ \ d_k(B)\big)}{|N_k(A)|}\right)^{-1}$$

lrd_k measures the density of a sample's region

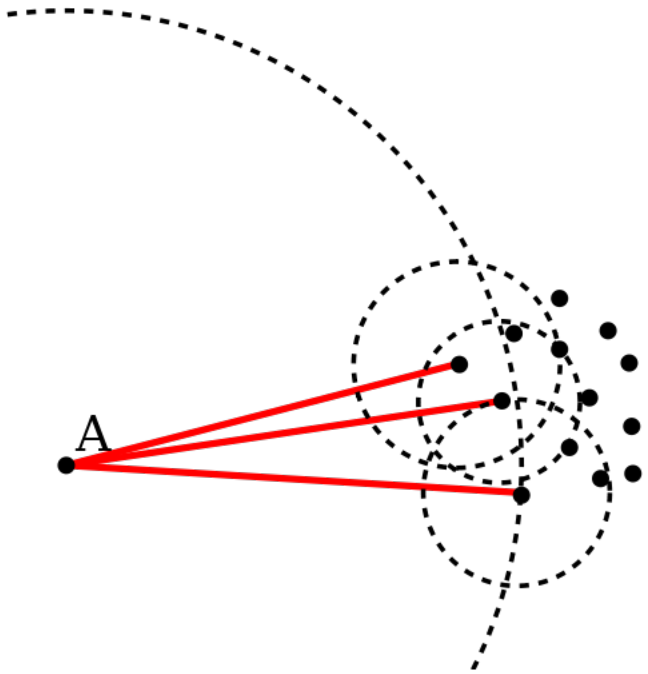
Local Outlier Factor (LOF) calculation

Definition of LOF_k:

Then we define the LOF:

$$LOF_k(A) = \frac{\sum_{B \in N_k(A)} lrd_k(B)}{|N_k(A)|\, lrd_k(A)}$$

$LOF_k(A)$ is the ratio between point A local density, $lrd_k(A)$, and the mean local density of its neighbors, $\frac{1}{k}\sum_B lrd_k(B)$.

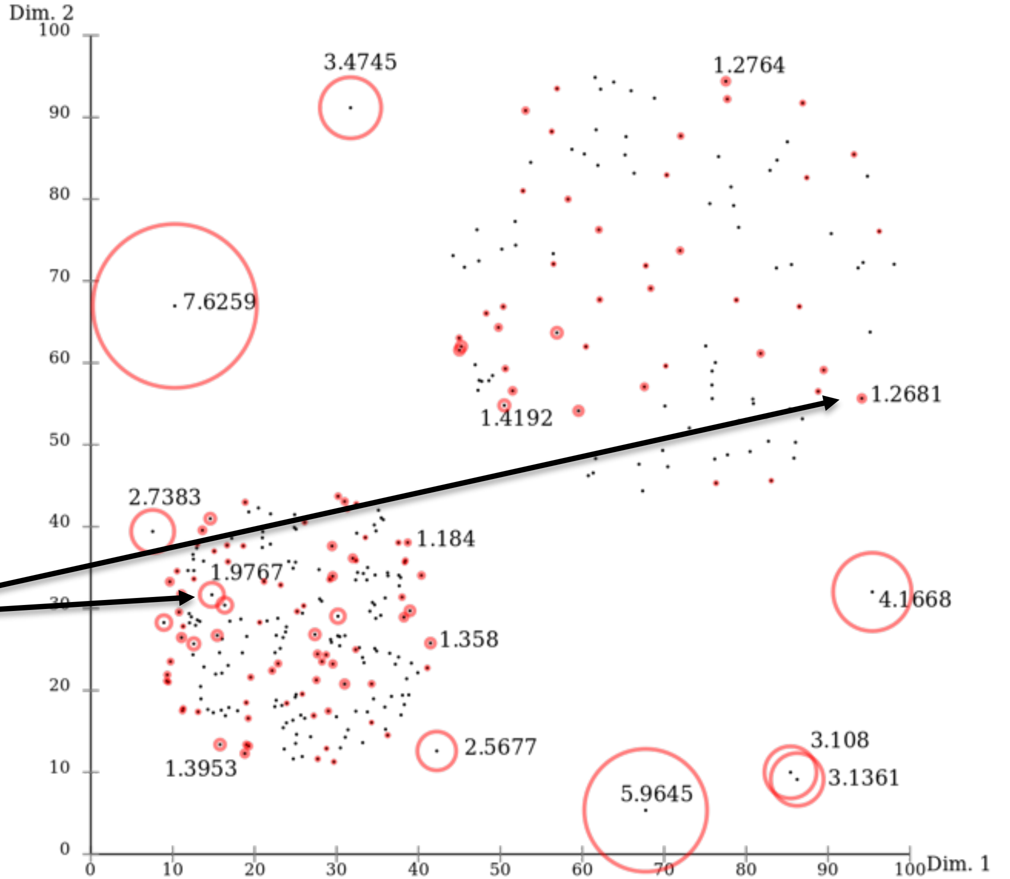


Local Outlier Factor (LOF)

LOF scores visualized:

Notice the differences in densities based on **locality.**

Points in high density areas might have higher score than points in lower density areas.



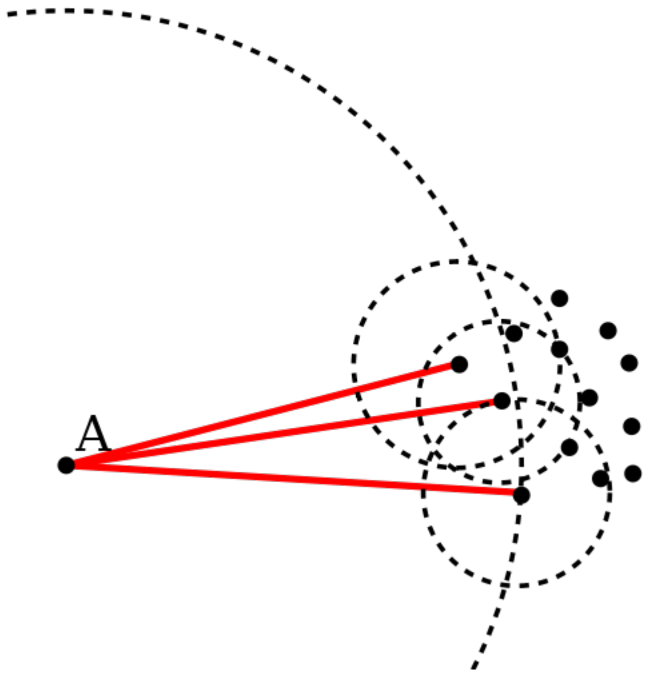
Local Outlier Factor (LOF) calculation

Q: What is the effect of the hyperparameter k?

A:

Low k: very local density calculation, high susceptibility to noise

High k: Averaging out over the entire dataset, no local effects



Local Outlier Factor (LOF)

< itc >

LOF implemented in scikit-learn as **sklearn.neighbors.LocalOutlierFactor**

```
from sklearn.neighbors import LocalOutlierFactor

clf = LocalOutlierFactor(n_neighbors=20)

y_pred = clf.fit_predict(X)

X_scores = clf.negative_outlier_factor_
```

- fit_predict(...) returns **1 for predicted inliers** and -**1 for predicted outliers**
- negative_outlier_factor_ - actual "outlier" scores

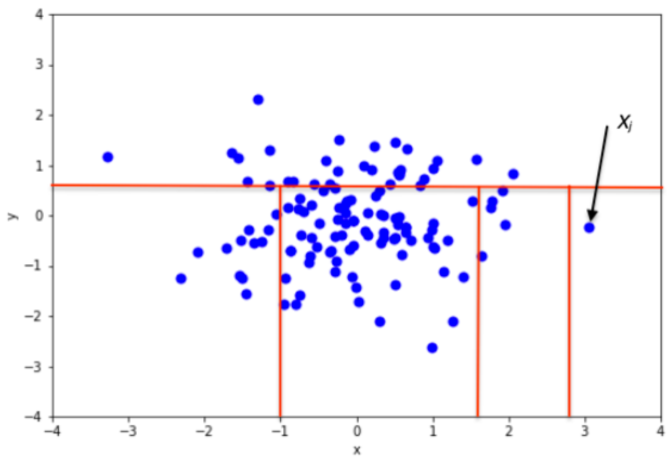
Notes:

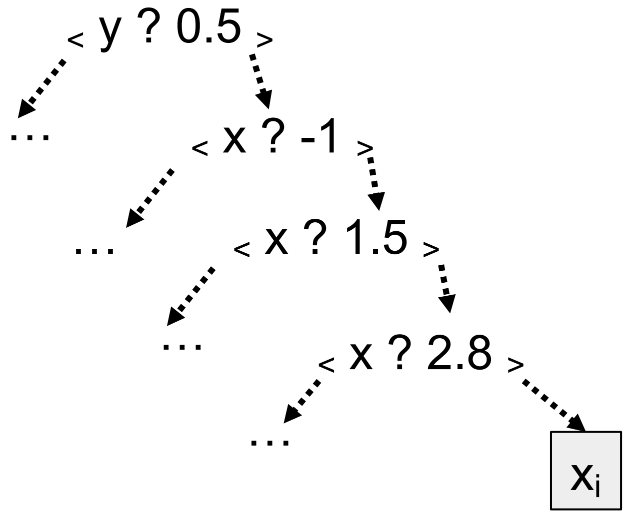
- negative_outlier_factor_ – since "negative" LOF, therefore: −LOF < -1 → outlier
- LocalOutlierFactor **has no** predict(...) **function** because it does not have a decision boundary (undefined on new data)

Isolation Forest – Tree Based

Isolation Forest

An ***isolation tree*** *is a* ***random*** *binary decision tree on the samples. In an isolation tree all leaves contain only one sample*





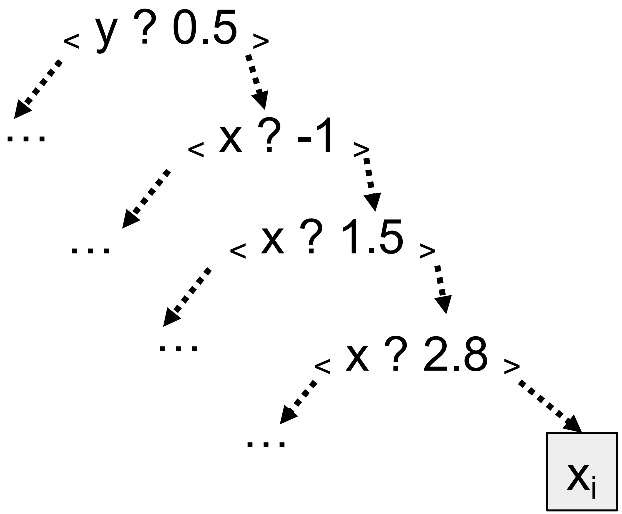
Isolation Forest

Each ***isolation tree*** is built by repeatedly picking a **random feature** and splitting it at a **random value** between its minimum and maximum values

h(…) : length of path to sample in isolation tree

Here $\boldsymbol{h(x_i) = 4}$

Samples with high ***h(…)*** value will be classified as outliers.

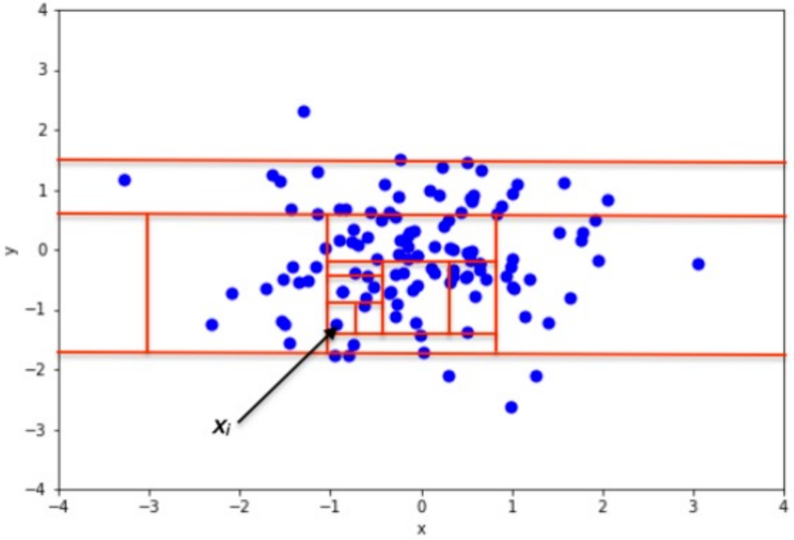


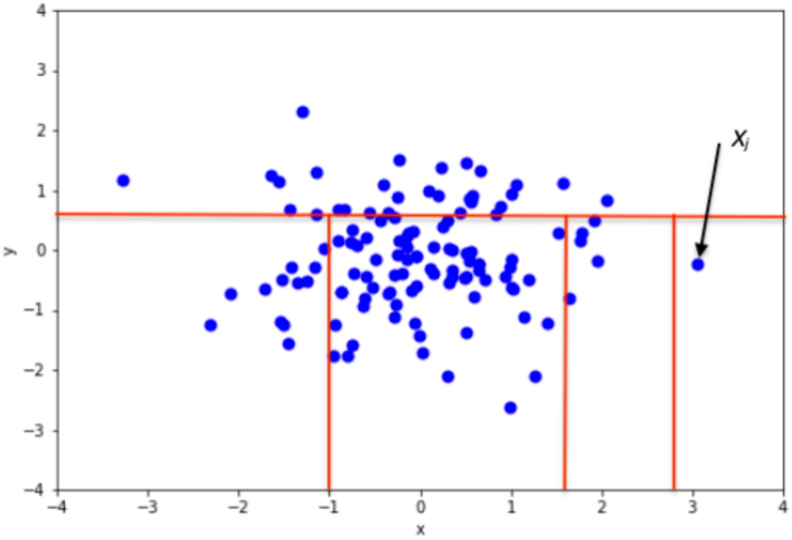
Isolation Forest

<itc>

Idea: On average, an **outlier** sample can be isolated with fewer "random" partitions of the feature space

- *Random* – random **feature** at random **value** of feature





Isolation Forest

Q: what is the problem with using one isolation tree?

A: it is random, by chance we might catch (or not) an outlier, or misclassify an inlier.

Q: How can we solve this?

A: Create many random trees and average their output

Isolation Forest

Implemented in scikit-learn as **sklearn.ensemble.IsolationForest**

```
from sklearn.ensemble import IsolationForest

clf = IsolationForest(n_estimators=100)

clf.fit(X_train)

y_pred_test = clf.predict(X_test)
```

predict(...) returns **1 for predicted inliers** and -**1 for predicted outliers**

You may also use decision_function(...) to get scores, with decision boundary at score=0

Isolation Forest vs. LOF

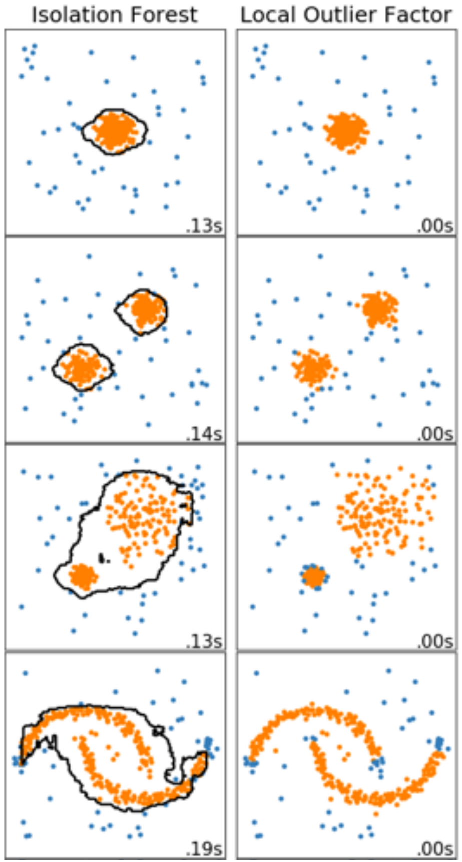
Isolation Forest:

- Gives **decision boundary** (more interpretable results)
- **“Global” outliers** - does not account for local density differences (example 3 to the right)
- Can be applied to unseen data

LOF:

- No decision boundary (less interpretable results)
- **Locality** - Better on outliers with **different densities** (example 3)
- K-NN calculation may be slow on large datasets
- Cannot be applied to unseen data*

Can be applied to unseen data when performing novelty detection, see [sklearn documentation]

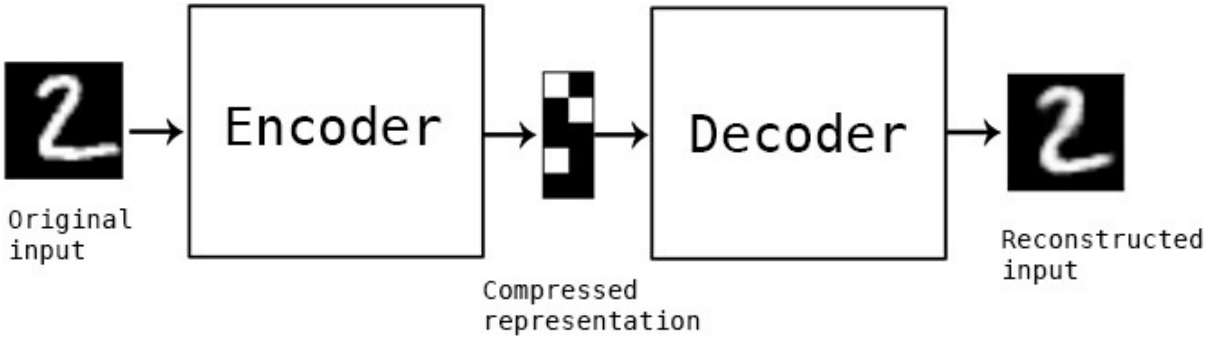


Anomaly detection with autoencoders

Anomaly Detection with Autoencoders

What is an autoencoder?

A **deep learning** model which learns a “compressed” encoding of a dataset so that it can be reconstructed:

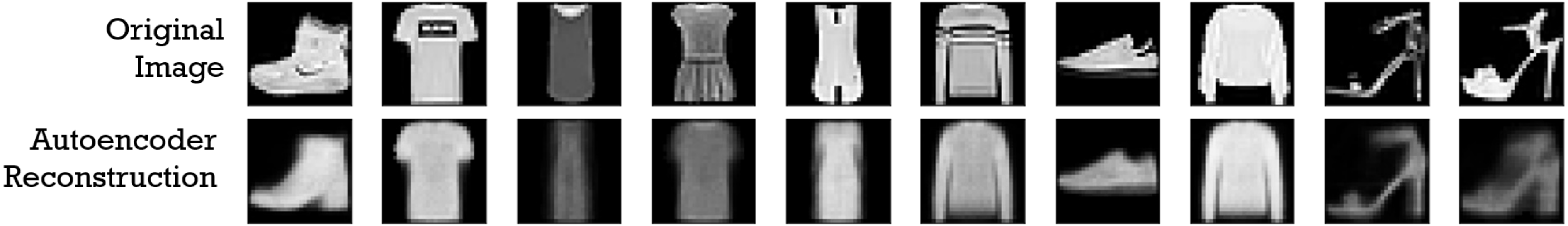


from https://blog.keras.io/building-autoencoders-in-keras.html

Autoencoders

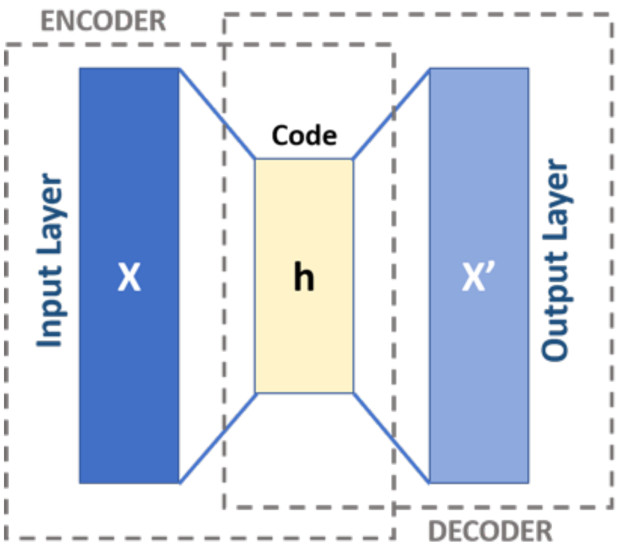
Autoencoders – can solve unsupervised learning problems by making them **self-supervised** (using the input data as its own target).

Can be used for **dimensionality reduction** – finding a good lower-dimensional representation of the data that preserves its important features.



Autoencoders

Autoencoder model architecture: "Forces" input data through smaller-dimensional hidden layer, and trains model to output something as close as possible to the input data



For **dimensionality reduction**, can look at **output of hidden layer** directly to get compressed representation of input

Anomaly Detection with Autoencoders

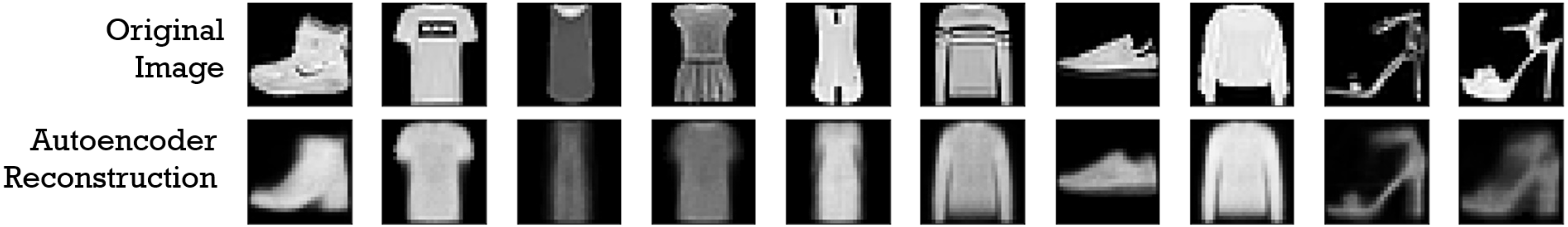
Idea of **anomaly detection with autoencoders**:

The learned compression scheme will be optimized for the type of data contained in “normal” data points, so it will **probably lead to information loss** on outliers.

Calculate error of every sample. What will be error of outliers compared to inliers?

Anomaly Detection with Autoencoders

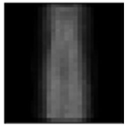
< itc >



Example: Autoencoder trained on images of clothing. If we input a digit from MNIST, the reconstruction would probably be very different from the input (**higher loss**)

Anomaly Detection with Autoencoders

 –  = low (inlier)

 –  = low (inlier)

 –  = low (inlier)

 –  **= high (outlier)**

Anomaly Detection with Autoencoders

Hands-on tutorial on implementing autoencoders in Keras:
https://blog.keras.io/building-autoencoders-in-keras.html
(Written by François Chollet, the inventor of Keras!)

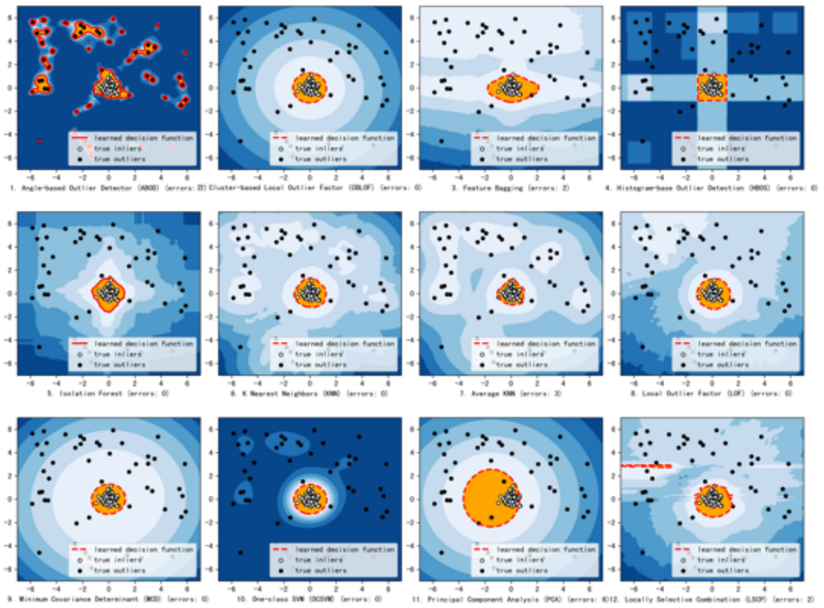
Example of anomaly detection using a Keras-based autoencoder: https://towardsdatascience.com/a-keras-based-autoencoder-for-anomaly-detection-in-sequences-75337eaed0e5

PyOD library

PyOD – Python Outlier detection - standalone library (not part of sklearn) with more than 30 Outlier detection algorithms.

Algorithms families in PyOD:

- Linear models – like PCA based
- Proximity-based – like LOF
- Probabilistic
- Neural Networks – like autoencoders
- Ensembles and combinations



Anomaly detection metrics

Anomaly detection Metrics

Are there metrics for unsupervised learning?

- Business stakeholder evaluation of small subset
- Exact metric:

  Labeling all / some anomalies and then using classification metrics:

  - Precision of anomalies
  - ROC AUC – FPR vs. TPR

Summary

- Motivation
- Intro to anomaly detection: types, challenges
- Local Outlier Factor (LOF)
- Isolation Forest
- Anomaly detection with autoencoders
- Anomaly detection metrics