Content:

- So far, you have learned many concepts that are there in Machine Learning
- The purpose of this lecture is to let you know how you can take the concepts that you already know like Random Forests, SVMs, KNN etc., and modify them for using them for other purpose rather than classification.

Introduction to Anomaly/Novelty/Outlier Detection

What is an Anomaly?

- Anomaly is nothing but synonymous of an **outlier**. These terms are often interchanged, and maybe called as **Novelty** depending on the context.

What's the difference?

- **Anomaly** means something which is not a part of the normal behaviour
- **Novelty** means something unique, or something that you haven't seen it before(novel)
- Anomaly/Outlier/Novelty Detection is used almost everywhere in almost any industry because of the fact that these points can make huge losses for a company, or they can highly impact their systems which can give misleading results.
- Anomaly/Outlier/Novelty Detection is highly used in Finance industries. Major applications are to detect Fraudulent transactions via any source (online banking, credit card, etc.)

Let's now see how we can detect anomaly/novelty/outlier.

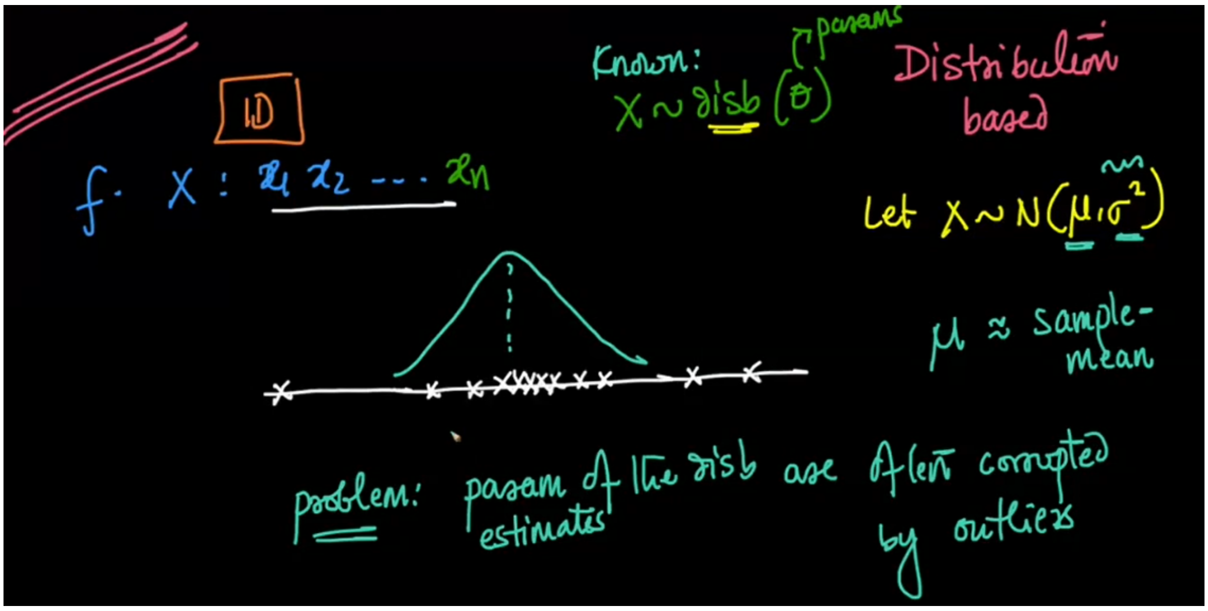
Distribution Based

The simplest way for detecting an outlier would be to use distribution parameters (mean and standard deviation).

- Consider a feature X with some observations $x_1, x_2, ...x_n$ with some outliers.

We know that X will follow some distribution D which will have parameters θ

- Let X follow a **gaussian distribution** with some mean(μ) and standard deviation(σ^2)
- If we know the distribution of the data, we'll try to fit the distribution. But, the problem arises with the distribution parameters.
- While we know the distribution, the parameter estimates of the distribution are often corrupted by the noise/outlier


1D
f: X : x_1 x_2 ... x_n
Known:
$X \sim disb(\theta)$
params
Distribution based
Let $X \sim N(\mu, \sigma^2)$
$\mu \approx$ sample-mean
problem: param estimates of the disb are often corrupted by outliers

How can we robuslty estimate parameters of the distribution?

- We do this using a algorithm called **RANSAC** which stands for **Random Sample Consensus**

Let's see what RANSAC is and how it works.

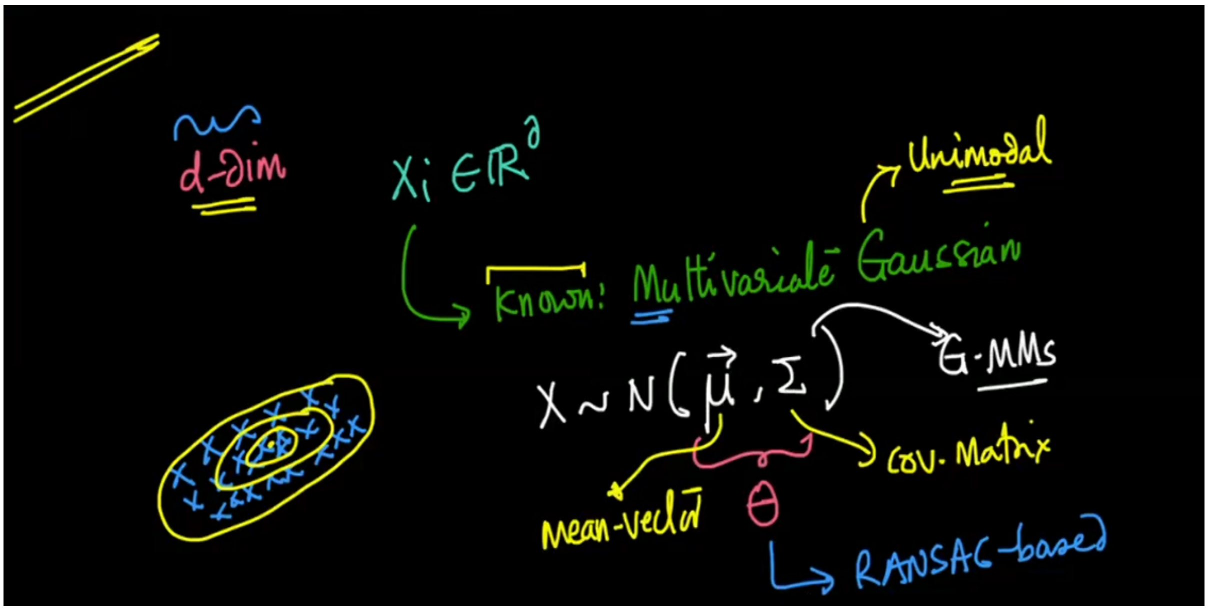
Random Sample Consensus (RANSAC)

- RANSAC is a **trial-and-error** approach which works very good in real life.
- Imagine a dataset X with n number of points having parameters μ and σ^2. Let's call them θ collectively
- There are mainly three steps involved in RANSAC. They are:
    1. Sample a subset of point from the dataset (n'). We consider this points as an inlier.
    2. Now, compute a model that estimates the parameters μ and σ^2 of the sampled points.
    3. Score the model which indicates how many points will support the model.
- We repeat these three steps iteratively and then select the model that is best supported by the data, which then, tells which points are inliers and which are outliers.

Extending the idea to higher dimensions

- Till now, we assumed that we have only a single feature X which was following Gaussian Distribution.
- Now, imagine we have a d**-dimensional** data X where each point $x_i \in R^d$ and the data is not labelled.
    - If we know the datapoints $x_i s$ follows multivariate gaussian distribution(unimodal), then X follows normal distribution;

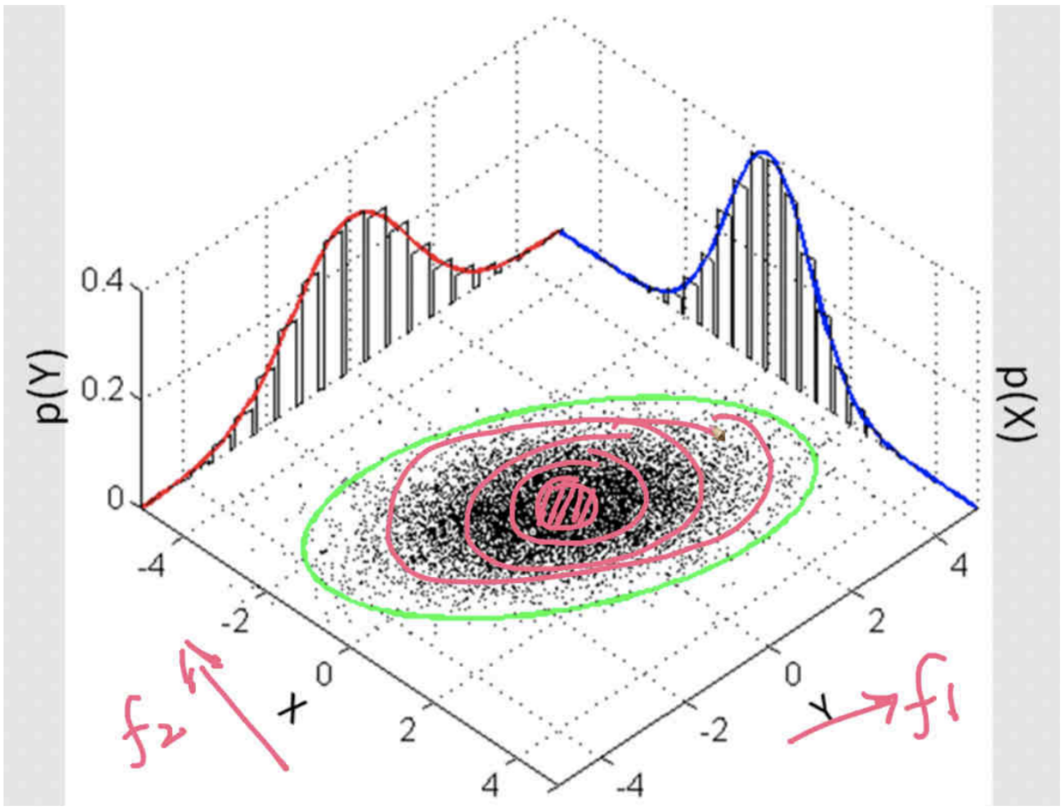
      $X \sim N(\vec{\mu}, \sum)$, where μ is mean vector and $\sum$ is a covariance matrix
    - Here we'll consider $(\vec{\mu}, \sum)$ as θ
    - If you remember from GMMS, in multi dimension space, the shape of gaussian was similar to a hill where the density of the points was highest in the middle contour, and it keeps getting low as we move away from the center
    - In this case too, RANSAC can be applied. Farther away a point is from centroid, we'll know that it is an outlier.

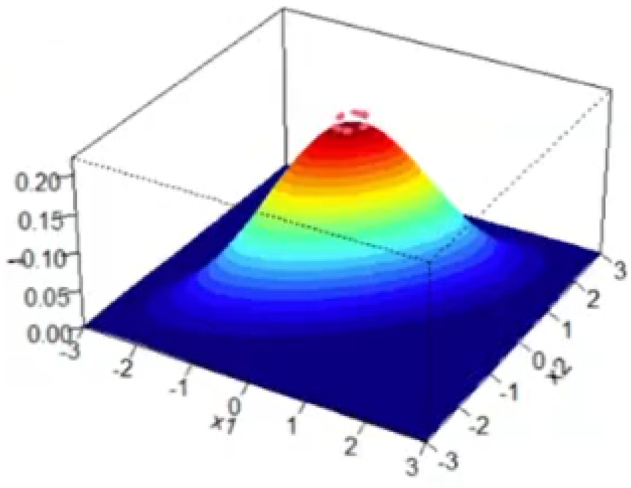


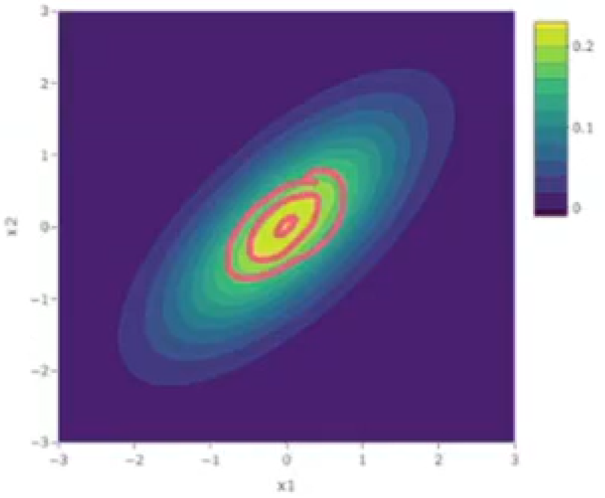
Similar principle is used in another method called **Elliptical Envelope**. Let's see what it is!

Elliptic Envelope

We know that a Unimodal Multivariate Gaussian Distribution on a single plane will look like ellipses if visualized on a plane. This idea can be extended to find out an outlier







- Given some data X where $x_is \in R^d$ and X follows Normal Distribution being unimodal, Elliptical Envelope robustly estimate the parameters $(\vec{\mu}, \Sigma)$.
- The term robustly means without getting impacted by outliers
- Next, then we remove the points that are outliers which are very far away from the centroid

Till now, we've assumed that the points are following Gaussian Distribution, that leads to a question.

So, for non-gaussian, do we need to convert into gaussian distribution?

- While the elliptical envelope method makes an assumption that the distribution is gaussian, the strategy can be applied to any distribution

- As long as we know any distribution and its parameters θ, we can extend our strategy to use RANSAC and estimating parameters θ
- The can be other distributions such as multivariate poissons, multivariate log normals, etc., but we dont use them as much
- One other strategy is to convert them into gaussians but we dont necessarily have to.
- As long as there is any distribution we can calculate the probability of $x_i \in X$ using PMF/PDF

sklearn walkthrough

Scikit-learn implements **EllipticEnvelope** as a part of **covariance** module. Let walkthorugh the parameters that are important

1. assume_centered: It is for assuming that that the data is centered at 0, i.e. $\vec{\mu} = 0$.

- By default, it is set as $False$. If set to $True$, it will just estimate the covariance matrix.
- It uses FastMCD approach, and not RANSAC approach. MCD refers to **Minimum Covariance Determinant**
- FastMCD performs in a similar way of RANSAC where it takes a subset of points, and using them it tries to estimate the distribution parameters robustly.

2. support_fraction: It tells how many points to use to estimate the parameters $(\vec{\mu}, \Sigma)$

3. contamination: It says what percentage of our data do we think are outliers.

- It takes values from 0 to 0.5, where 0.5 represents that 50% of our data is noisy. Default assumption value is $0.1 \sim 10\%$

Disadvantages

You might have find the concept of elliptic envelope quite straightforward, but there are limitations.

- It cannot be used non-unimodal data
- It is specifically for multivariate gaussians

If the data fails to meet the assumptions of unimodal and multivariate gaussian, the whole things crashes.

Next up is Isolation Forest. Let's see what that method has to offer!

Isolation Forest

Consider a dataset D which contatins datapoints $x_1, x_2 \ldots, x_n$ Just like Random forests, Isolation Forests builds many trees. Let's see the intition behind Isolation Forest

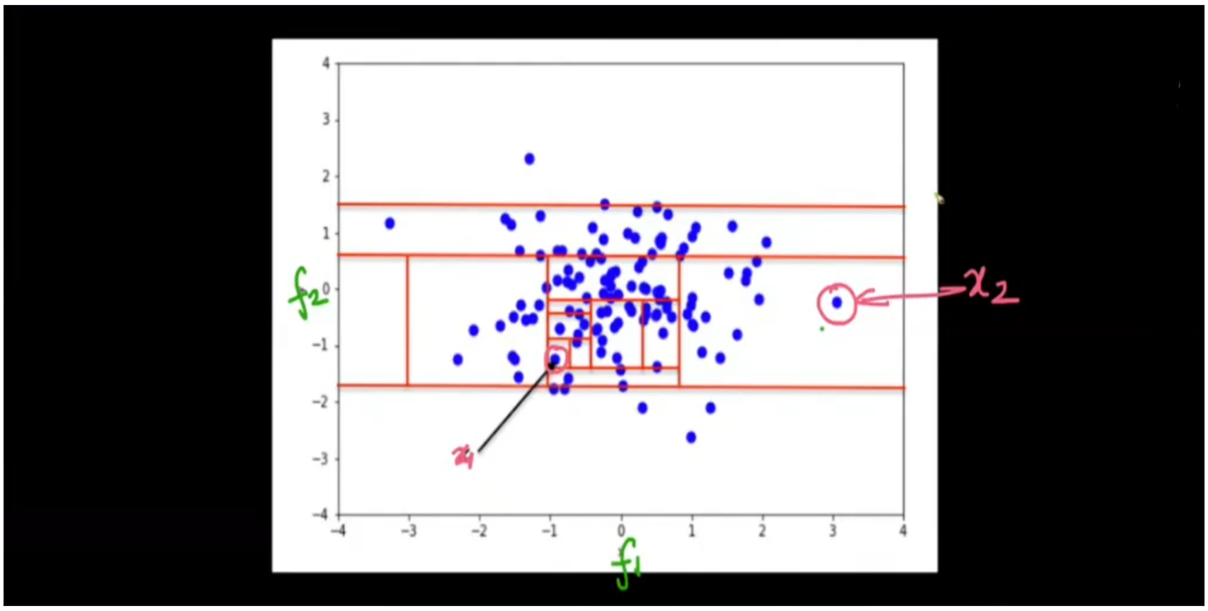
Intuition

These are the stpes involved in Isolation Forest:

- Build many trees like random forests
- For each tree:
    - Randomly pick a feature
    - Randomly threshold that feature
    - Build each tree until the leaf consists of only one datapoint

Isolation Forests are also known as **iForests**

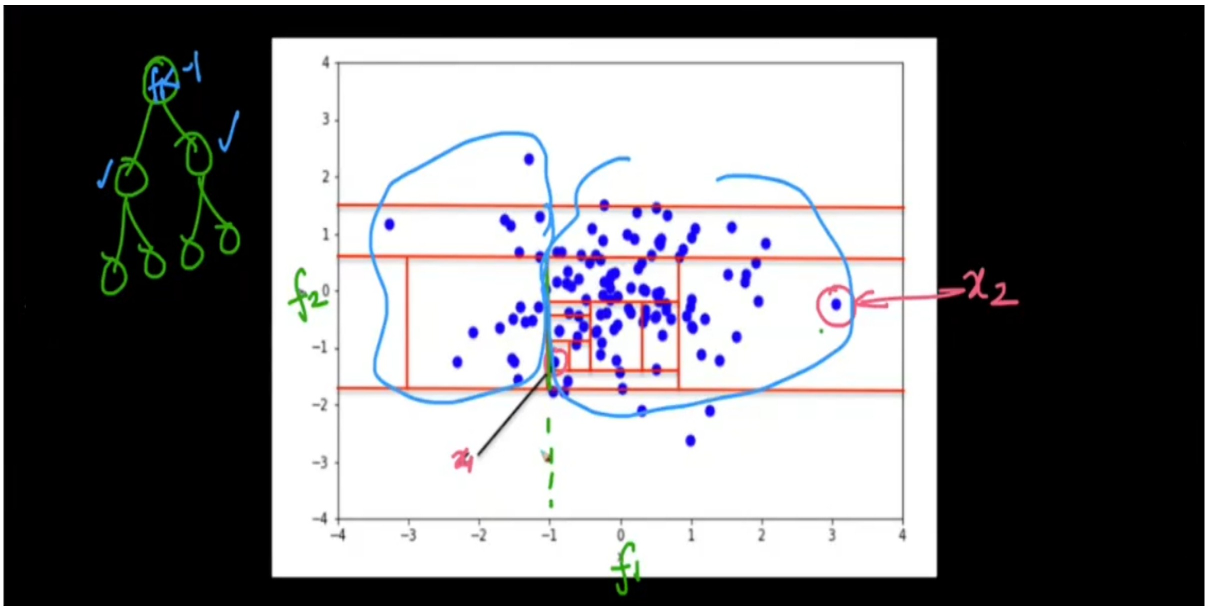
Consider the plot along feature f_1 and f_2 given below



In isolation forests, we are building totally random trees. So if we pick feature f_1 and put a threshold there will be a vertical bar.

Similarly, if we pick feature f_2 and put a threshold there will be a horizontal bar.

For example, if we pick feature f_1 and we select threshold as $f_1 < -1$, then our first root node will be based on this condition



Based on the diagram below, on an average, between x_1 and x_2 which node will be at more depth?

- The node containing x_1 will be at more depth.
- Observe that the point x_1 is in dense region, and point x_2 is far away
- That is because, to break the point x_1 from all the other points, more and more splits will be required and that will increase the depth of the node containing point x_1

So, to sum it up, the idea behind Isolation forest is,

- On an average **outliers** have **lower depth** in the random trees
- On an average, **inliers** have **more depth** in the random trees

Evaluation of Isolation Forest

How can we evaluate Isolation Forests?

- Imagine, we have build 100 random trees. For each point x_i in the dataset, we can get an average depth.
- We use this average depth to convert into a metric.
- Apart from this, there are lot of different metrics, that people have came up with over the years
- But, the basic intuition is that lesser the average depth, higher likelihood is there that it is an outlier

But, how do we decide average depth for a point to be classified as an outlier?

- There is no one metric specifically used for average depths in iForests. At the end, whichever metric you use, it is based on the threshold.
- There are a lot of metrics that researchers have came up with over the years.

But, what if the number of datapoints is large? Wouldn't it mess up the Isolation Forests?

- iForests can be made on subset of samples.
- We use this subset as train dataset and the rest of the data as test dataset

Sklearn Walkthrough

We can implement Isolation Forest with the help of sklearn's **IsolationForest** method present in **ensemble** module.

Lets see some of the parameters that **IsolationForest** expects

1. n_estimators: It represents the number of base learners. By default, the value is set equal to 100

2. max_sample: It is the number of samples to extract from the dataset to build the trees(row sampling). By default the value is set to **auto** and sklearn picks reasonably a good figure for iForests

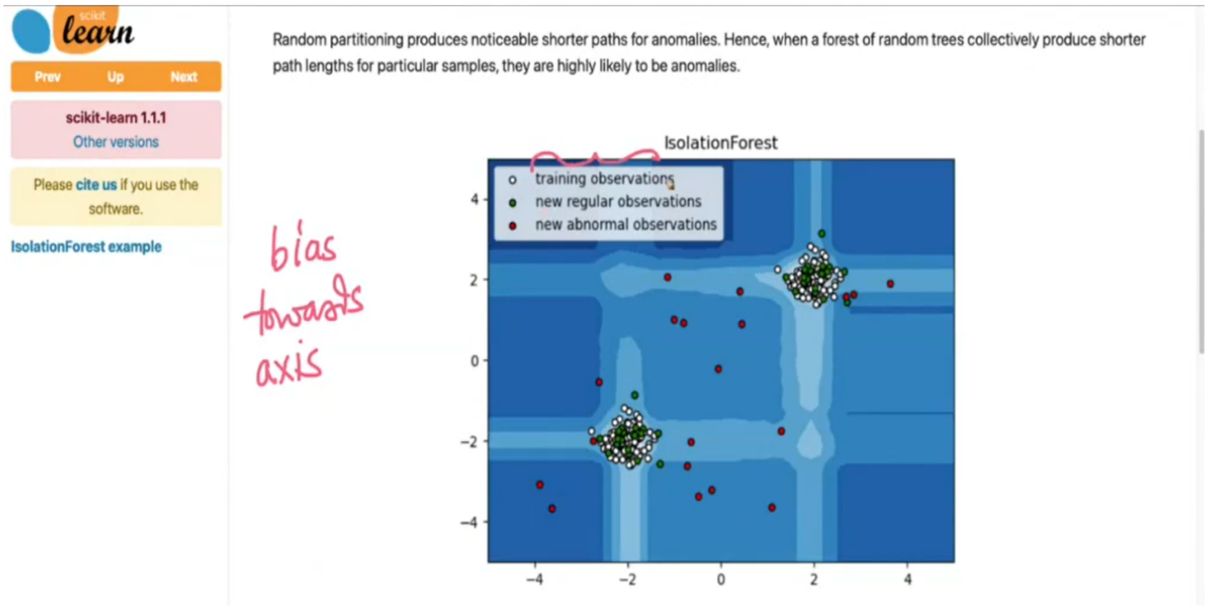
3. contamination: It tells the proportion of outliers in the data. The range is between $[0, 0.5]$

4. max_features: It is the number of features to extract from the dataset to build the trees(column sampling).

Isolation Forests are heavily used when dataset size is very large because algorithms like Elliptical Envelope are typically harder and they have very strong assumptions

Major Disadvantage of Isolation Forest

- **One of the major limitation of iForests is that they are biased towards axis parallel splits.**
- iForests makes splits and these splits are always parallel to either of the axis.
- Because of this, the boundary will not be smoothed.
    - In the diagram given below, the different shades of blue represents the likelihood of a point to be an outlier. Darker the color, it is more likely that the point in that region will be an outlier
    - We've trained iForest model using training data(white points)
    - It is tested on testing data(red + green) where red color indicates outliers

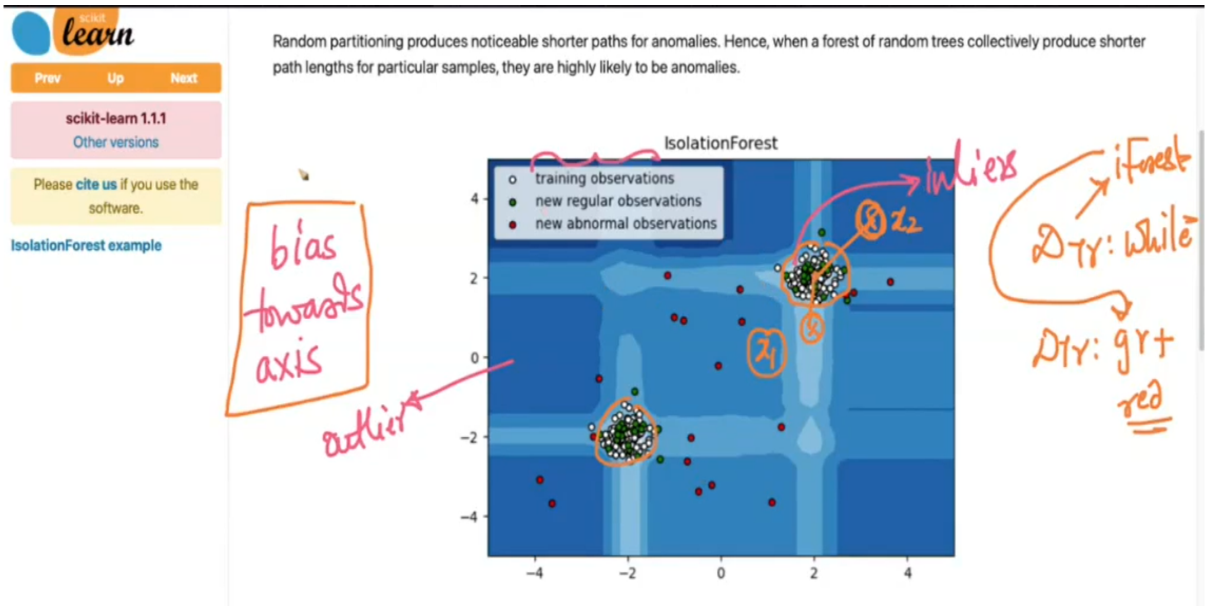


Random partitioning produces noticeable shorter paths for anomalies. Hence, when a forest of random trees collectively produce shorter path lengths for particular samples, they are highly likely to be anomalies.

Now imagine two points x_1 and x_2. x_1 as shown in the diagram given below.

- Both the points are almost equidistant from nearest cluster. x_1 is on the axis and point x_2 is off axis.
- Because the model is biased towards axis, it will classify point x_1 as an inlier and x_2 as an outlier

This is also known as banding in signal processing



Random partitioning produces noticeable shorter paths for anomalies. Hence, when a forest of random trees collectively produce shorter path lengths for particular samples, they are highly likely to be anomalies.

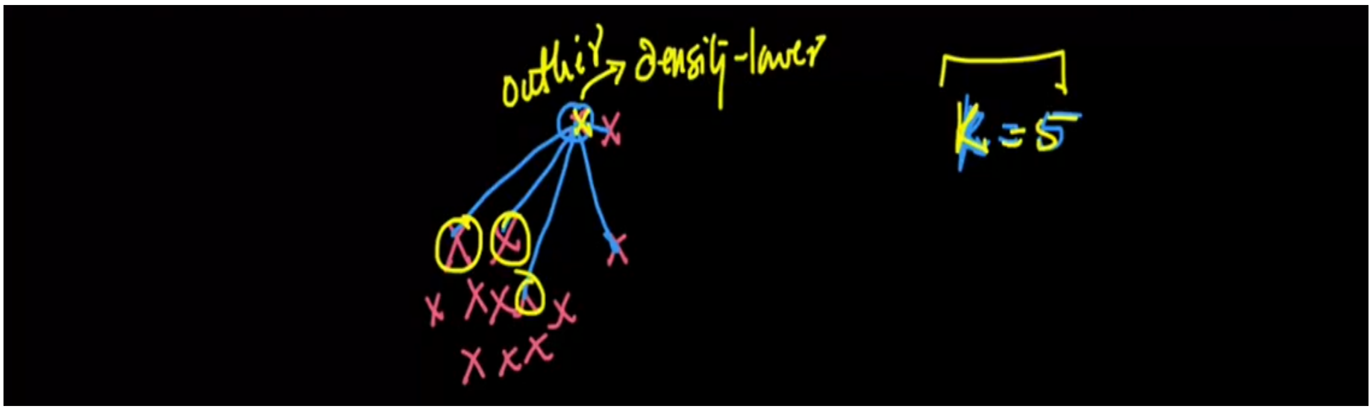
Local Outlier Factor (LOF)

- On a higher level, LOF is based on two ideas: **KNN** and **density**
- The core idea behind LOF is to compare the density of a point with its neighbors' density

- If the density of a point is less than the density of its k neighbors, we flag that point as an outlier

Intuition

Imagine a bunch of datapoints as shown below



We compute the density of a point based on average distance.

- if average distane between a point and its k nearest neighbors is large, it is more likely that the point will be an outlier
- also, larger the value of k, more confident are the results

Moving forward, lets get familiar with some concepts to understand the working of LOFs

1(a) **K-distance**

- We define K-distance of a point A as the distance of point A to its k^{th} **nearest neighbor**
- In general, larger the value of k-distance is, farther away the points is from other datapoints

1(b) **Set:** $N_k(A)$

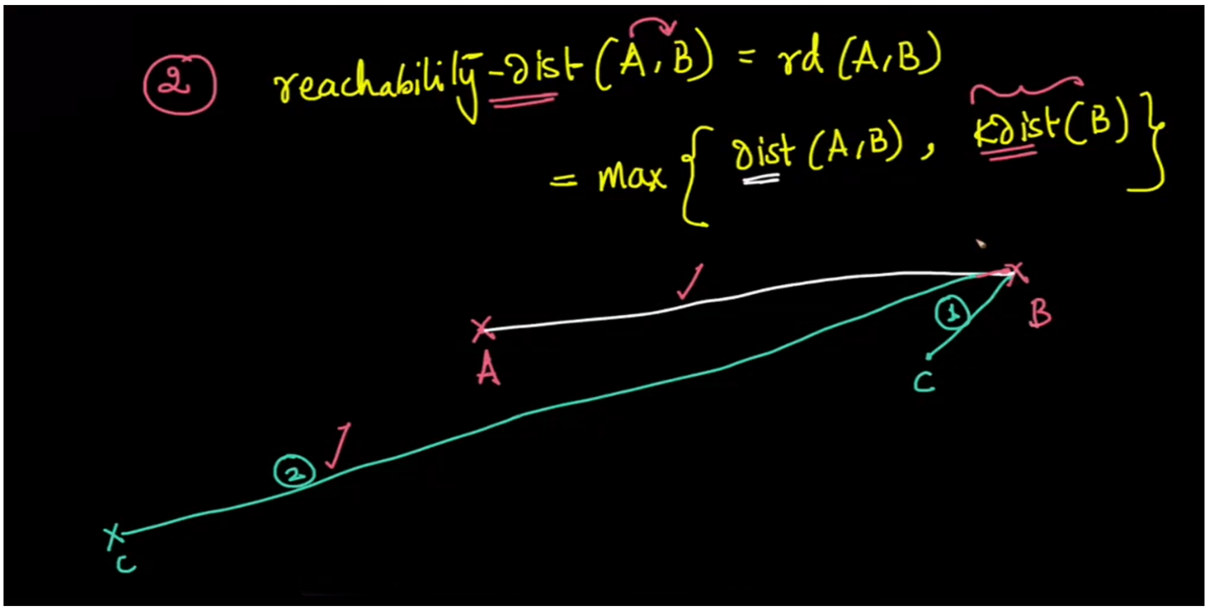
- It is a set of k-nearest neighbors of point A

2. Reachability distance

- From point A to point B, we define reachability distance as **maximum** of the distance from point A to point B and the maximum k-distance of point B

$$reac - dist_k(p,o) = max[dist(A,B), kdist(B)]$$

- Consider point B with some k nearest neighbors shown in the diagram below.


2
reachability-dist (A, B) = rd (A, B)
= max { dist (A, B), k-dist(B) }
A
B
C
1
2
C

- There is a possibilty that some neighbors might be close(condition 1) and some neighbors might be very far away(condition 2)
- In this case, there is a neighbor of point B whose k-distance $>$ the distance between point A and B, and hence, it is considered as its reachability distance
- In this case, point A may or may not be a k-nearest neighbor of point B

3. Local Reachability Density

It is often represented as $lrd_k(A)$, which tells local reachability density of a point A

It is defined as the average reachability distance between point A and k neighbors

So, $lrd_k(A) = \dfrac{1}{\dfrac{\sum_{B \in N_k(A)} rd_k(A,B)\$}{N_k(A)}}$

- The summation in above equation represents the sum of reachability distances from a point A and set of neighbors B as $B \in N_k(A)$

Putting all together

We define Local Outlier Factore of point A as follows:

- $LOF_k(A) = \dfrac{\sum_{B \in N_k(A)} lrd_k(B)}{|N_k(A)|.ldr_k(A)}$

Whoa!!! This looks scary. What does it mean? Lets see.

- $ldr_k(A)$ is nothing but the density of point A
- The expression $\dfrac{\sum_{B \in N_k(A)} lrd_k(B)}{|N_k(A)|}$ is the average neighborhood density
- So, LOF of point A is nothing but the average neighborhood density(lrd) of point A divided by the density of A

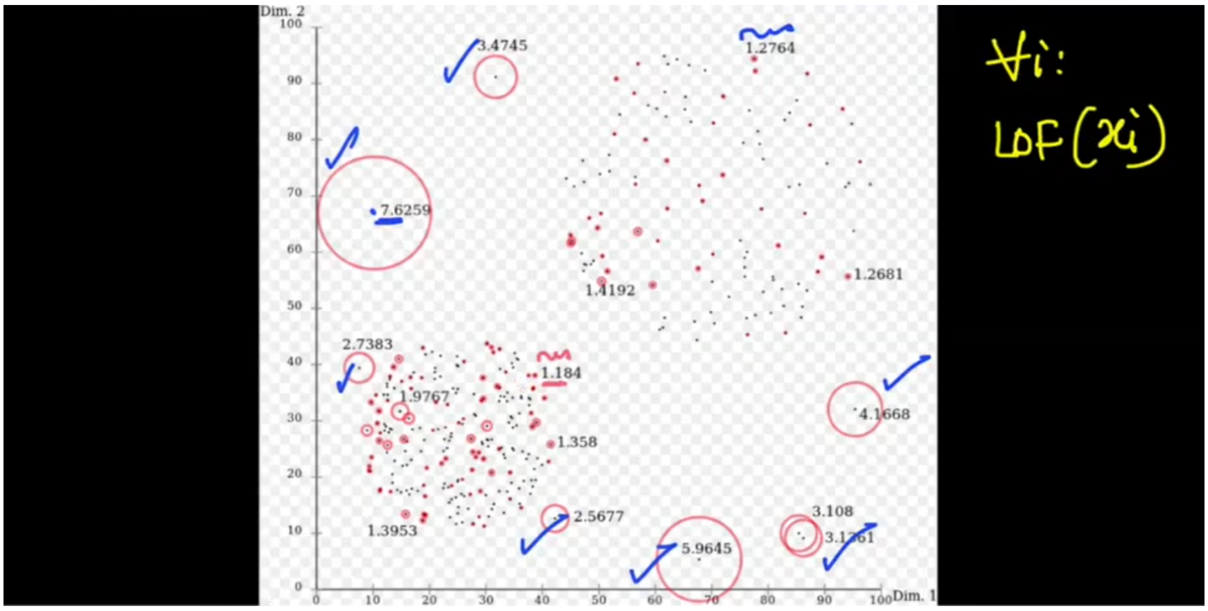
Interpretation of LOF

Now that we know how to compute LOF for a given point, questions is how to interpret it? Let's see.

- If $LOF(A) = 1$, then we can say that the point A has same density(lrd) as its k nearest neighbors
- If $LOF(A) > 1$, then the k neighbors of point A have higher denstiy than point A.
  - That does not mean point A is an outlier. It may or may not be.
  - But if $LOF(A) >> 1$, then point A is definitely an outlier.
- If $LOF(A) < 1$, then point A has more denstiy than its k nearest neighbors.

We compute the LOF for each point, and based on that extract the outliers

Let's now see how the results look like:



- You can see that extreme outliers have very high LOF values, while the border points of a dense region has slightly higher values than 1
- We dont mark these border points as outliers.
- There is a point with LOF value = 2.7383. We mark it as an outlier because it is certainly a lot higher 1.
- These is one limitation of LOFs. We don't know what threshold to use and it often depends on the use case and is decided by the domain expert

Disadvantages of LOF

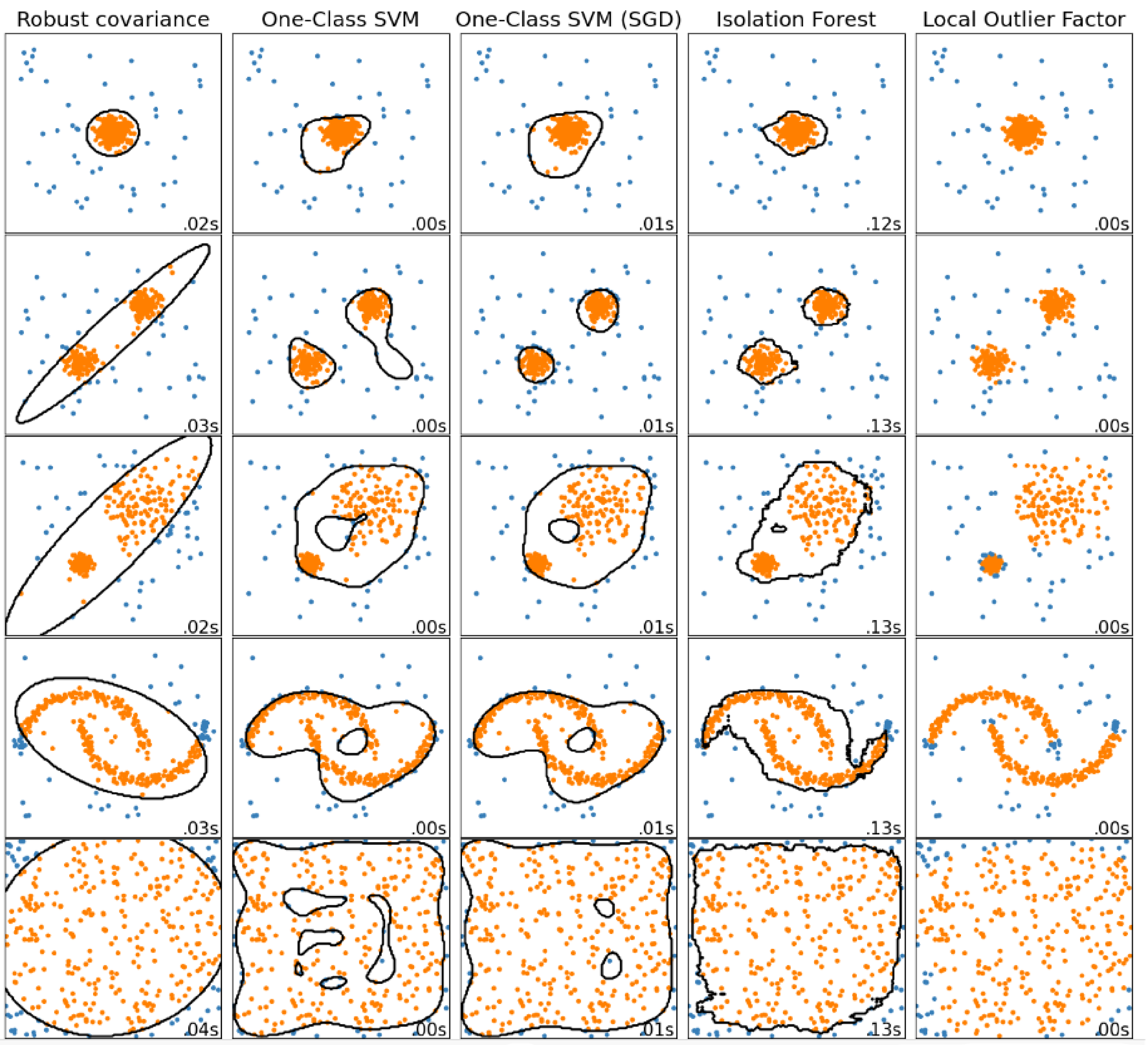
1. Finding optimal K
2. Finding threshold.
    - If LOF(A) >> 1, what is the threshold??
3. Cannot handle high dimensional data efficiently
4. High Time Complexity

Comparision of Methods

We'll now see a brief comparision of all these methods.

https://scikit-learn.org/stable/auto_examples/miscellaneous/plot_anomaly_comparison.html#sphx-glr-auto-examples-miscellaneous-plot-anomaly-comparison-py

The below given results were obtained from a toy dataset. Let's try to see where each methods performs well and where they might fail



Case 1: When data is dense at a certain region and sparse around it

- From row 1, We can see that all the methods that we studied performs relatively well when the data is dense at certain region and sparse around it
- You can observe that decsion boundary of One-class SVMs are a little bit broader than the cluster region which might include some outliers if present in that region

Case 2: When there are multiple dense regions in the data

- Here also, we can see that each methods performs relatively good.
- In elliptical enevlope(robust covariance) method, you can see the elliptical shapes being formed as a decision boundary. This can include some noise points if present inside the ellipse

Case 3: When one cluster is highly dense and other cluster is a little spread out

- Same problem arises with the elliptical envelope method as we saw in the previous case
- Other methods perform quite well and similar.

Case 4: When data has two archs

- In this case, you can see that the points in the outer parts of the archs are marked as outliers by all the methods, although they are a part of the archs

Case 5: When data is randomly spread

- All the algorithms marks the extreme points at all the four vertices as an outliers

Isolation Forests and LOF, both worked relatively well compared to other methods for all the cases, which explains why they are used oftenly in the industry

---