EEP 596: TinyML

Lecture 5: TinyML for Anomaly Detection

Dept. of Electrical and Computer Engineering

University of Washington

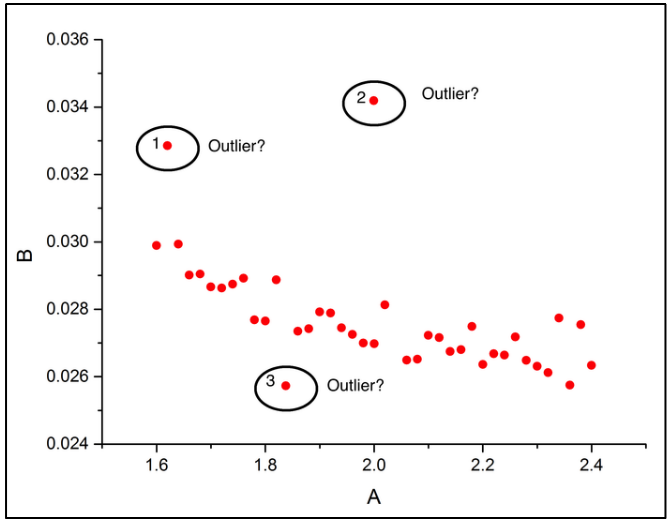
Instructor: Dinuka Sahabandu (sdinuka@uw.edu)

ELECTRICAL & COMPUTER ENGINEERING

UNIVERSITY of WASHINGTON

What is Anomaly Detection?

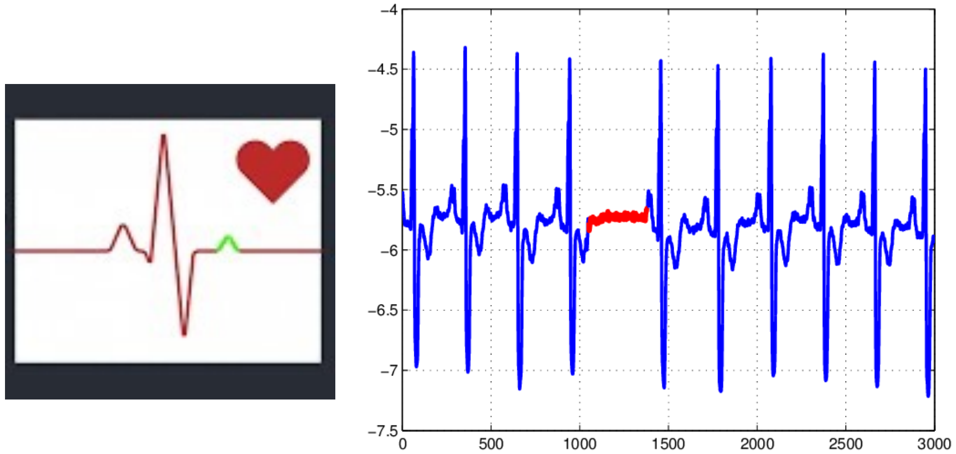
- What is abnormal or anomaly?
- Why detect an anomaly as it occurs?
- For what applications can we predict anomalies?
- How can TinyML or ML on the Edge **flag** and **help to perform preventive maintenance** in machine failures, metal fatigues that lead to cracks, and wear and tear of manufacturing assembly before failure happen?
- What type of ML algorithms are useful to raise the anomaly flags?
  - Supervised or unsupervised?
  - Classify with decision or Clustering?



Outliers in Data

Sensors for fall detection

Applications of Anomaly Detection



Electrocardiograms for heart rate monitoring

Factory machinery fault detection

Credit Card Fraud Detection

Detecting suspicious login activity

Anomaly Detection - Sensor Networks

Health

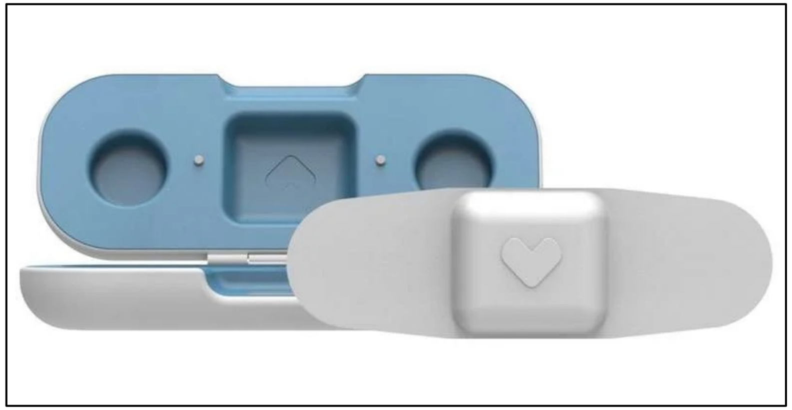
Industry

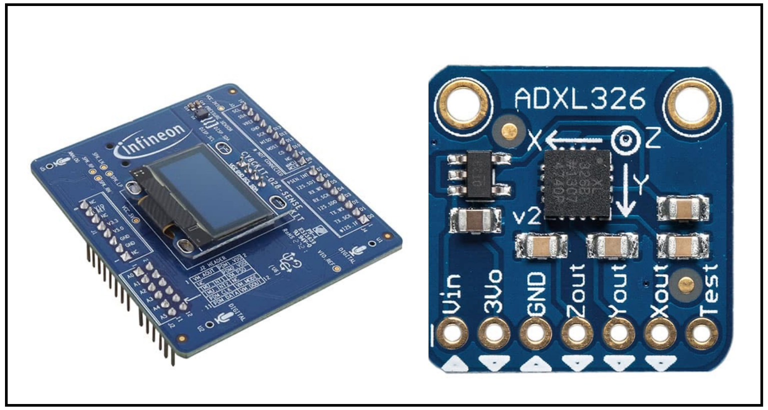
Security

ECG Sensor

Accelerometer

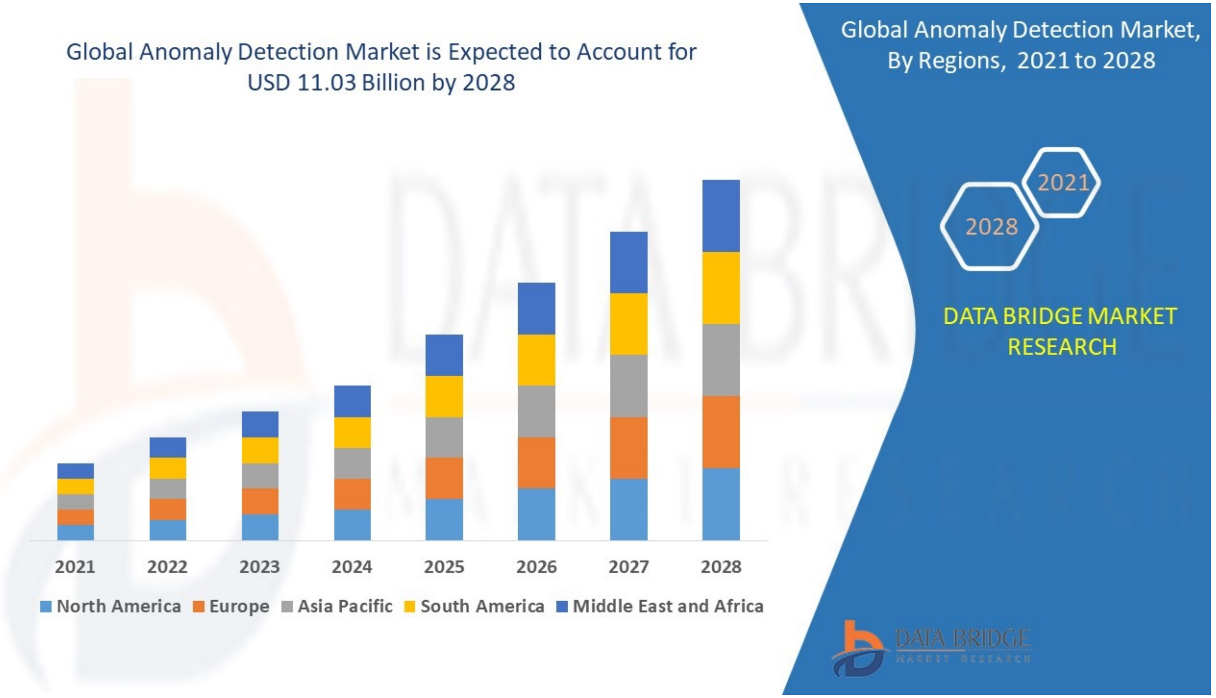
IR Motion Sensor

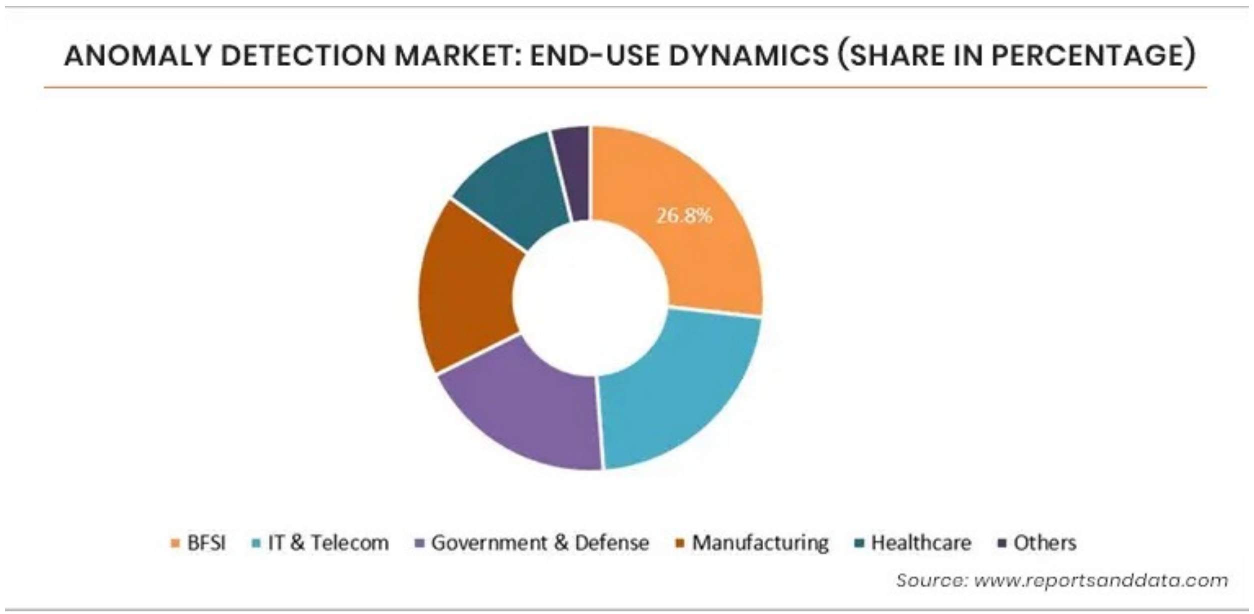




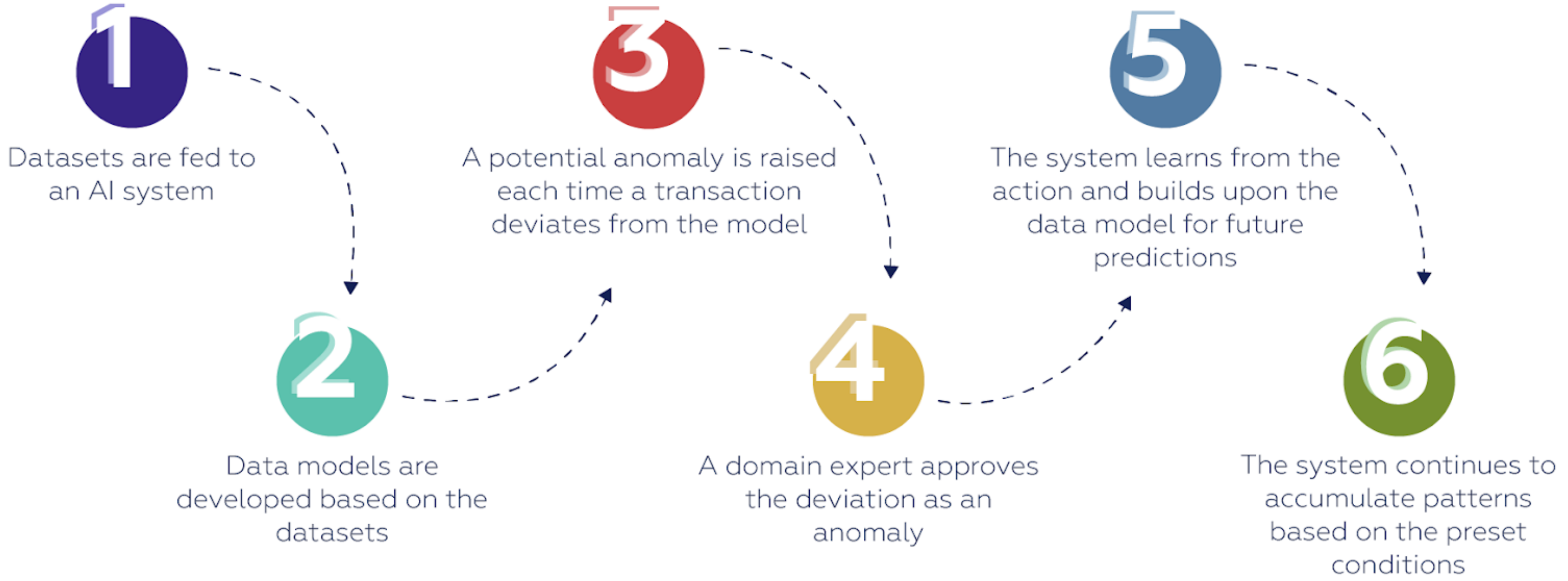
Anomaly Detection - Market Trends

- The growth of anomaly detection is spread across different market sectors - Banking, Financial Services and Insurance (BFSI) is the leading industry.
- Prediction for anomaly detection market expects exponential increase in adoption.





ML-4-Anomaly Detection - Process

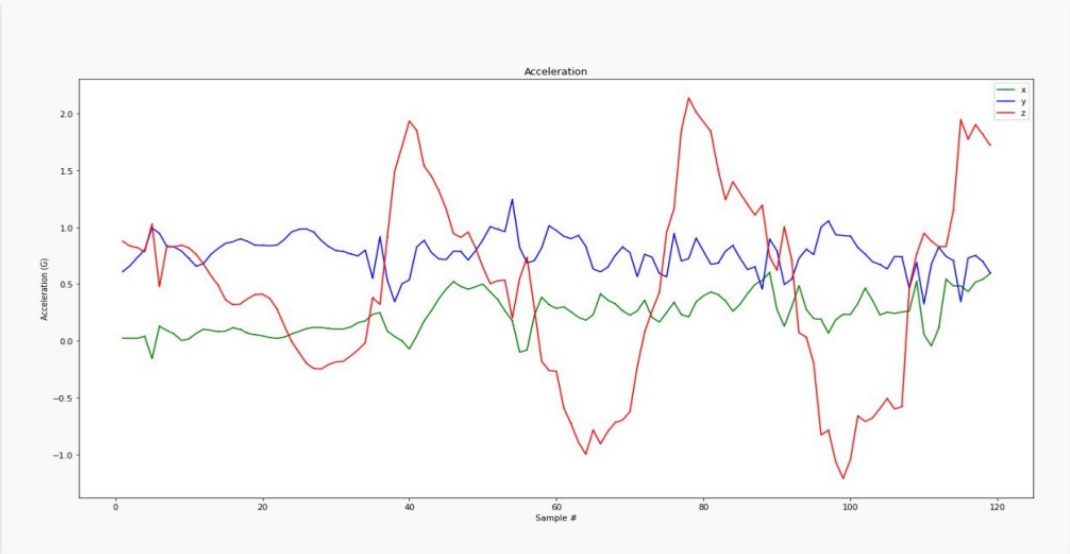


Source: https://www.kdnuggets.com/2019/10/anomaly-detection-explained.html

Anomaly Detection System

The ML model determines whether the machine/ equipment is prone to failure in the near future based on anomalies in current functioning.

Machine parts



Accelerometer and gyroscope data

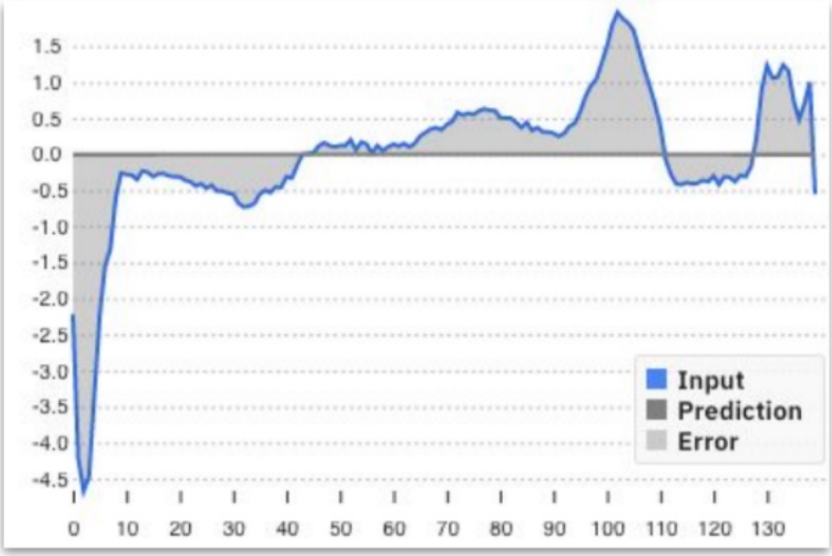
Prediction: Bearing likely to fail after ~2 weeks

Model Prediction

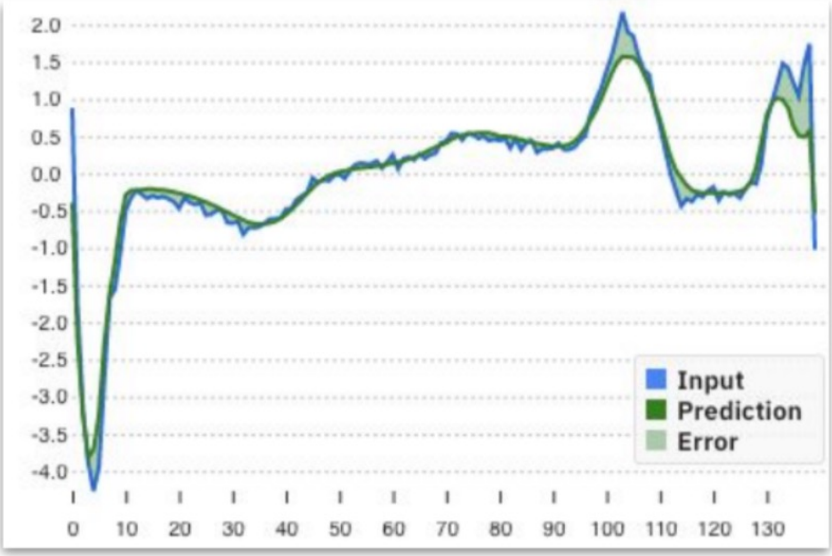
Anomaly Detection - Parameters

- Input – Observed data indicating current working condition/ status.
- Prediction – Model prediction for future behavior based on detected anomalies.
- Error – Difference between predictions and actual working.

Poor Prediction



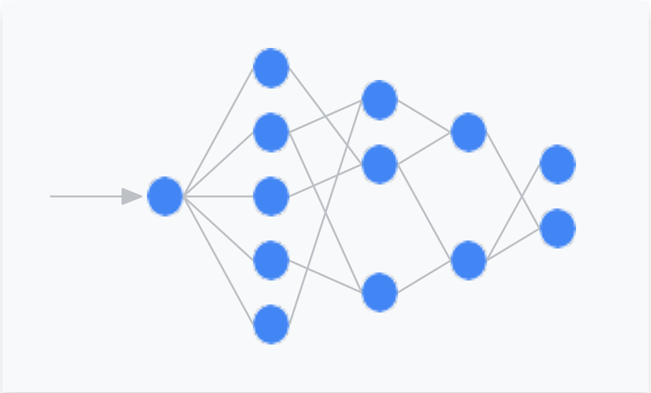
Good Prediction

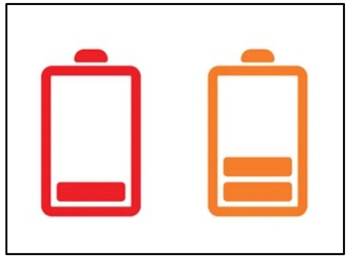


Source: https://drive.google.com/file/d/1SIfkL-3c5k0xGTQhJZg-s-Tv8T73nPuW/view

Challenges in implementation

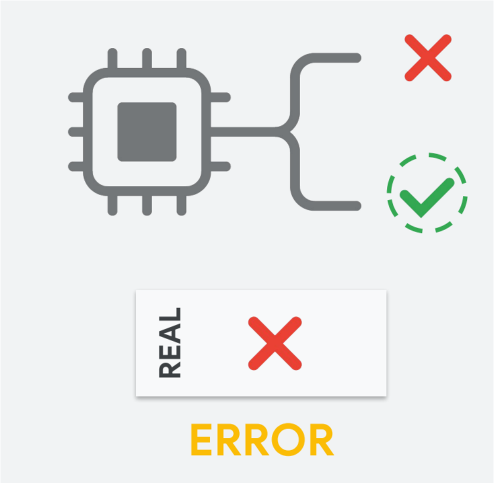
- On-device Computing is necessary since streaming to the cloud is expensive. For on-device computing, battery life is crucial.

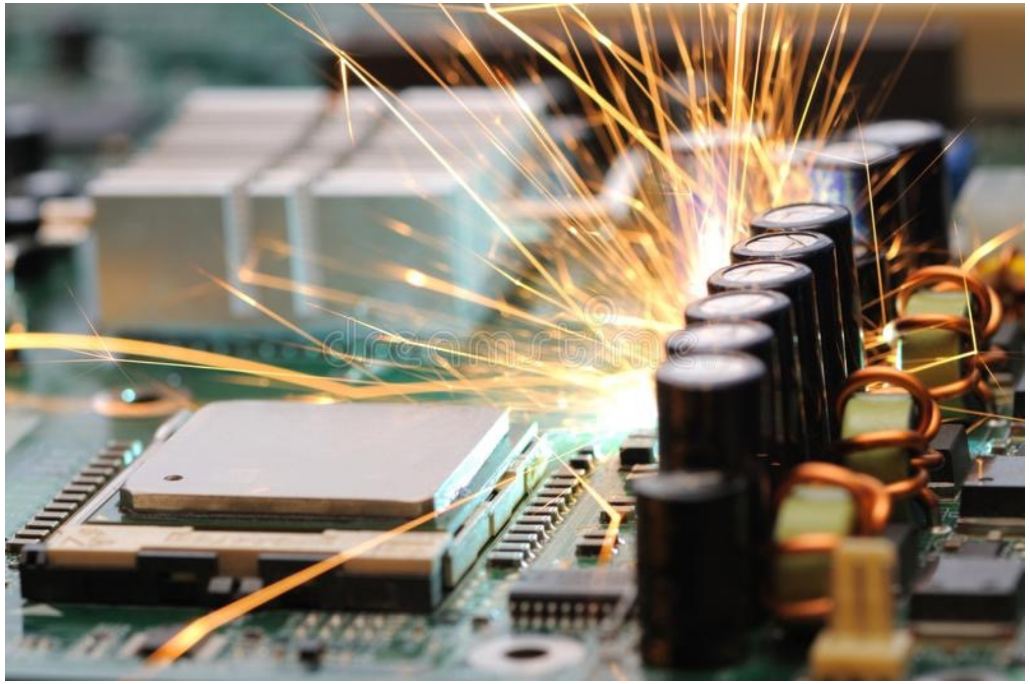




Challenges in implementation

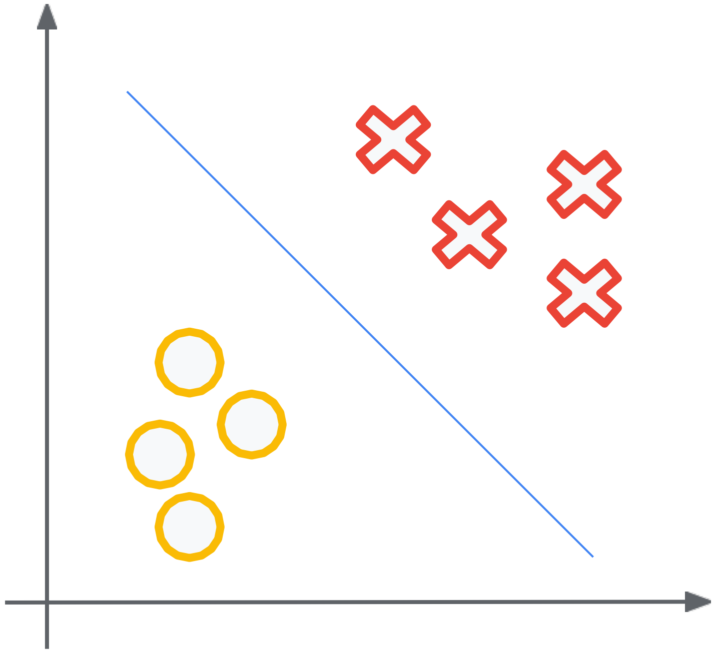
- False Negatives cause damage to devices and sometimes can result in catastrophic incidents.
- False Positive have a high-cost impact involved in manual inspection and component replacements.



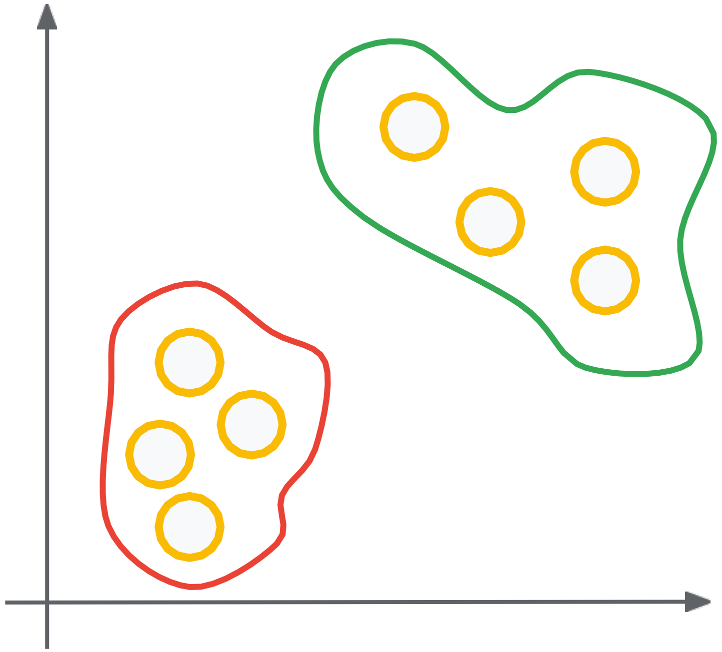


Supervised vs. Unsupervised Learning

Supervised Learning: labelled datasets (**decision and classification**)
Unsupervised Learning: unlabeled datasets (**clustering**)



Supervised Learning

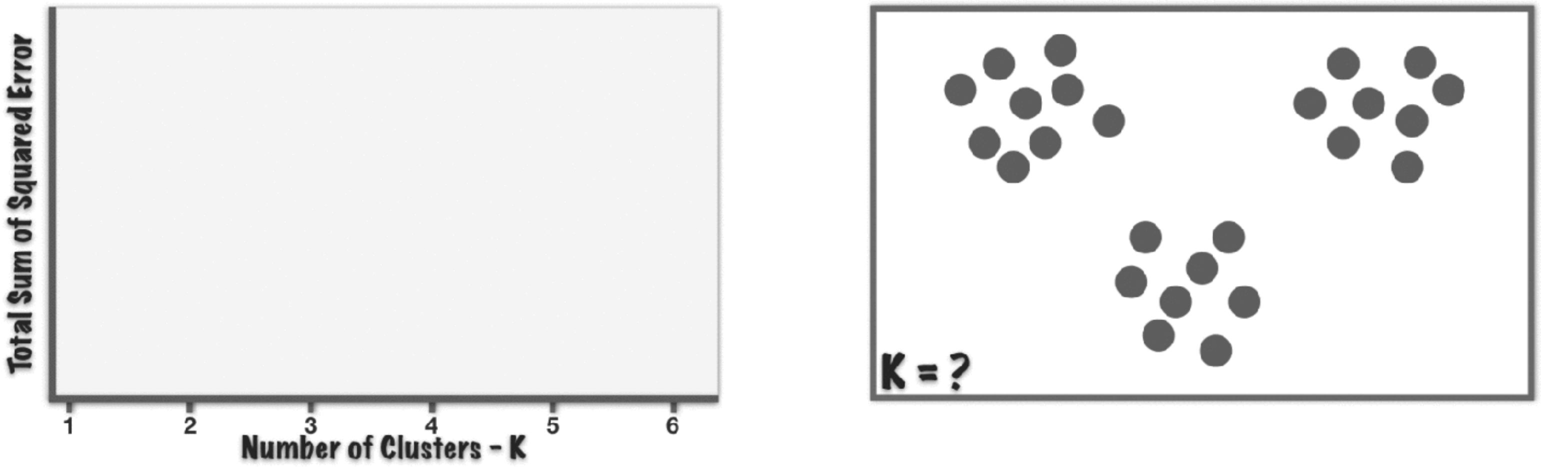


Unsupervised Learning

K-Means for Clustering

Introduction to K-means Algorithm

- Unsupervised Learning (No label required!)
- A **cluster** refers to a collection of data points aggregated together because of certain similarities.
- K-means looks for a fixed number (k) of clusters in a dataset.

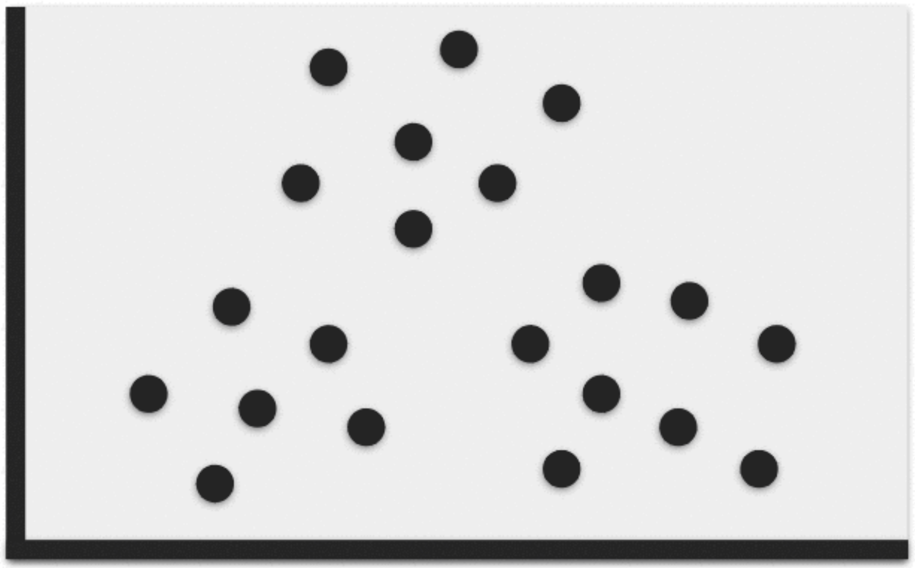


Source: https://towardsdatascience.com/k-means-a-complete-introduction

K-Means

How does K-means Algorithm Work?

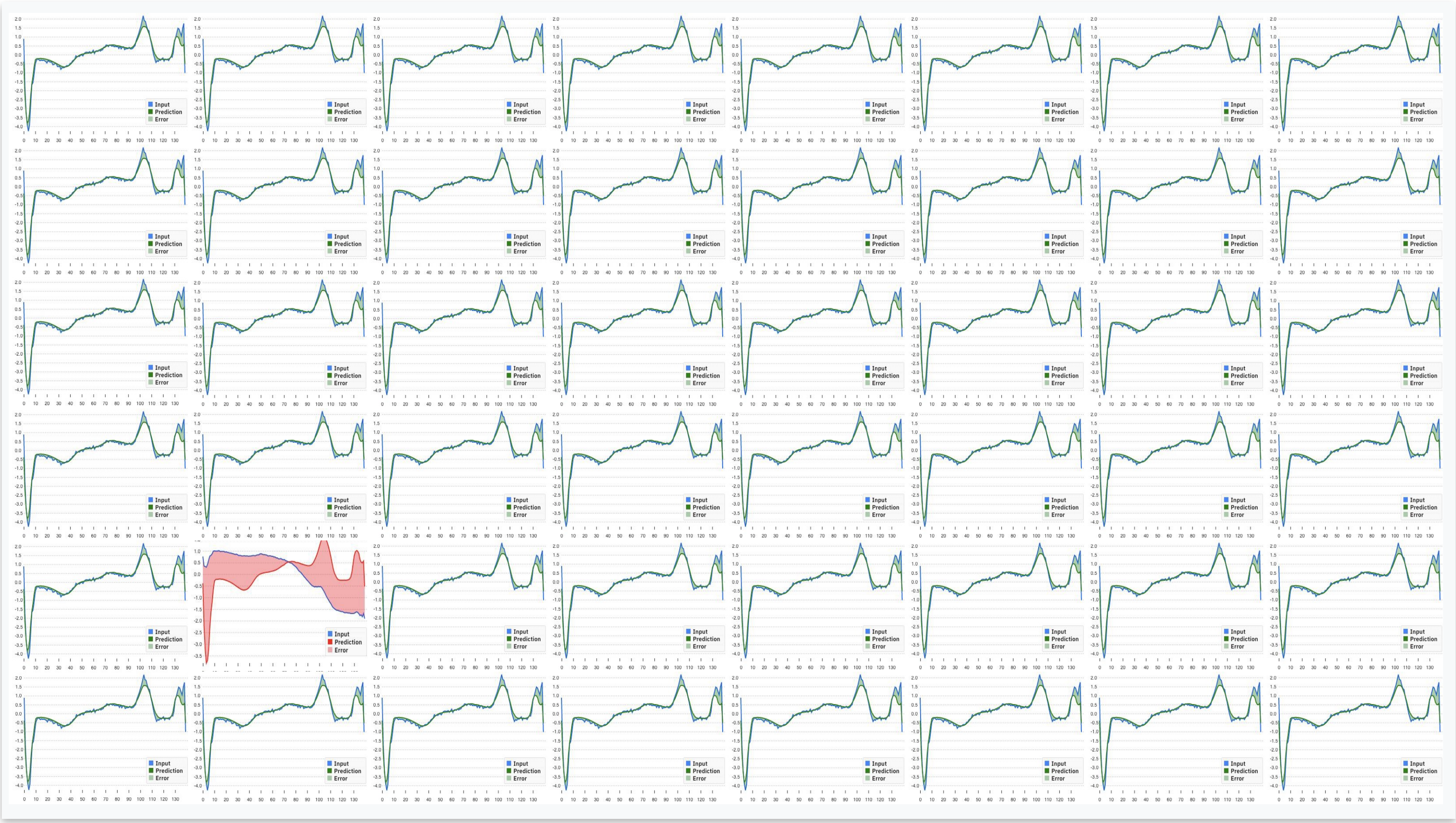
1. Randomly choose an initial *centroid* (center coordinates) for each cluster.
2. Until convergence do
   - Assign Step: Assign each observation to its nearest center
   - Update Step: Update the centroids as being the center of their respective observation.



Source: https://towardsdatascience.com/k-means-a-complete-introduction

K-Means Algorithm for Anomaly Detection

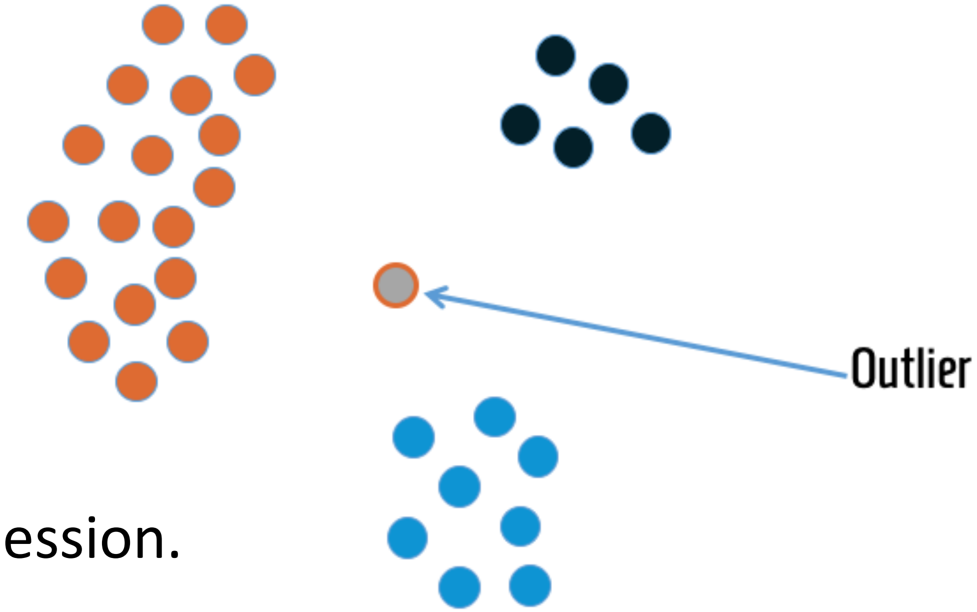
- Detect the outlier points in the dataset that should not belong to any cluster (e.g., use of Mean squared error > Threshold value to detect outlier)



K-Means Algorithm for Anomaly Detection

After clustering, anomalies can be identified by

1. The small clusters with less than a threshold (1% of total number of data points)
2. Isolation data points not belong to any cluster
3. A data point belongs to a cluster with more than 2 standard deviations (i.e., 95% confidence).

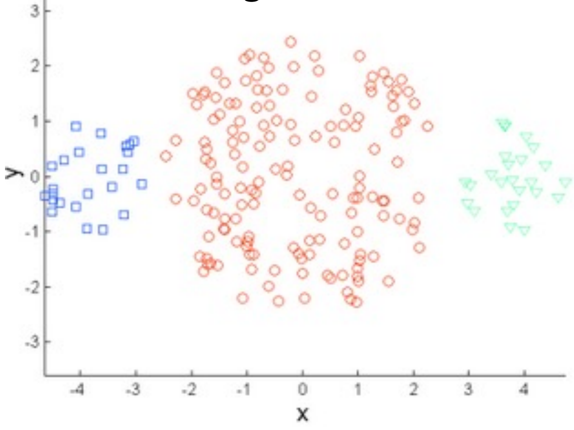


We will show more in lab session.

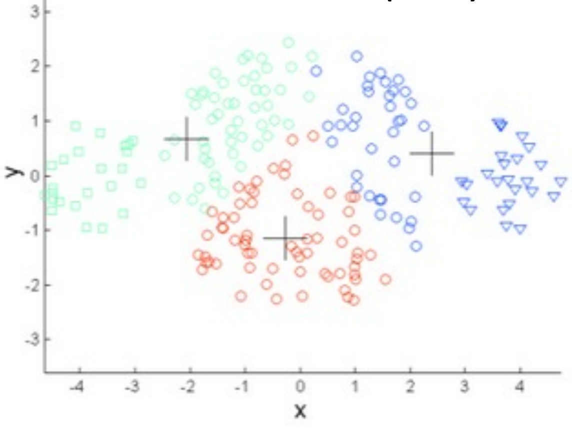
Challenges with K-Means

Clustering datasets with inherent clusters of different sizes

Original Data

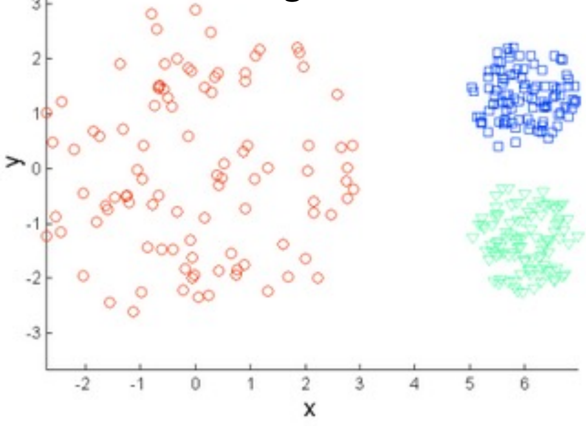


K-Means (K=3)

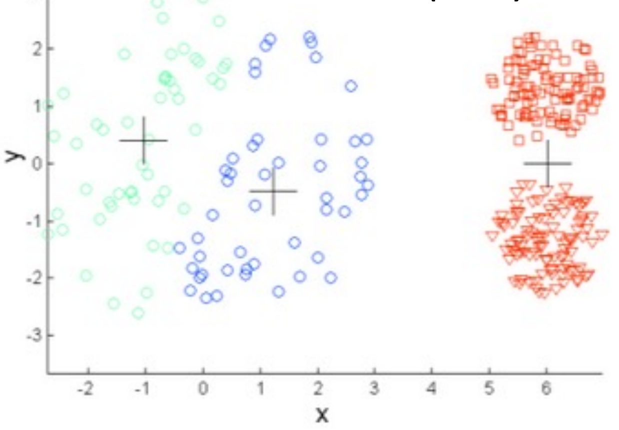


Clustering datasets with inherent clusters of different densities

Original Data

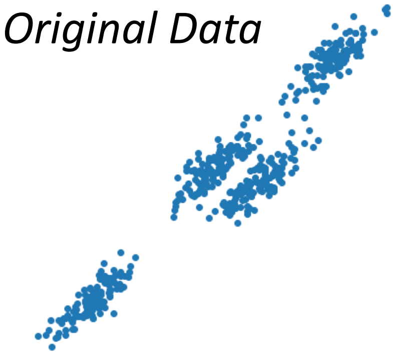


K-Means (K=3)

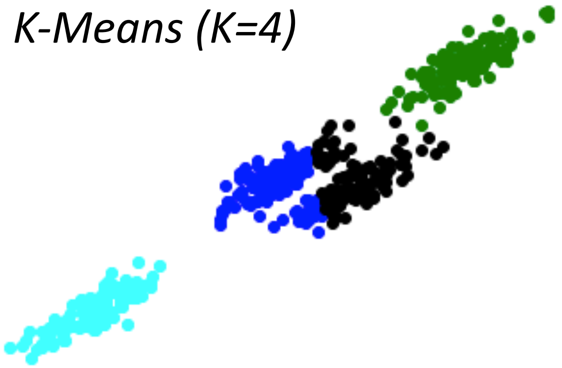


Clustering datasets with inherent clusters whose data points are non-uniformly distributed around the centroids

Original Data



K-Means (K=4)



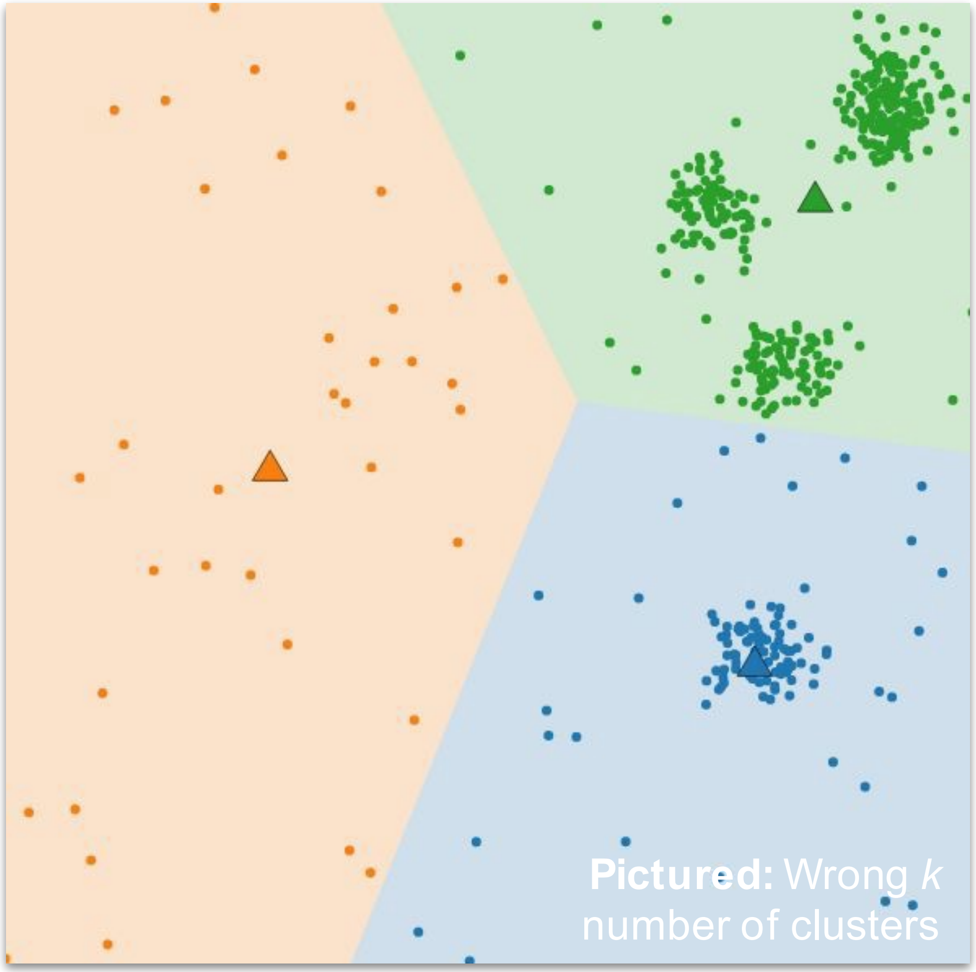
https://www.analyticsvidhya.com/blog/2019/08/comprehensive-guide-k-means-clustering/


5 min Break

Dimension Reduction

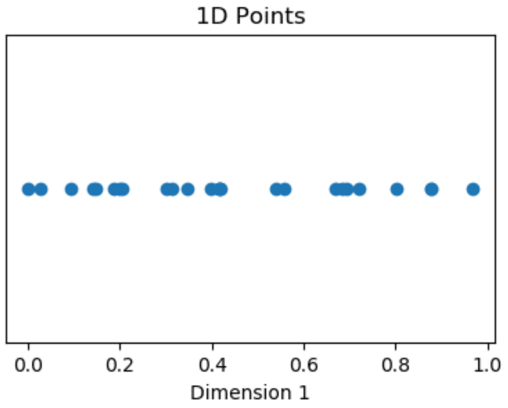
When may K-Means fail?

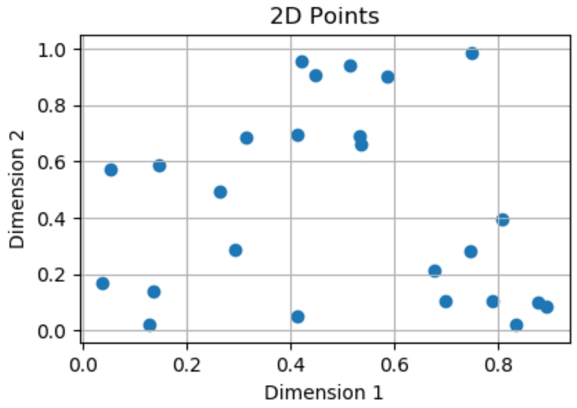
- Scales poorly with **large numbers** of observations
- **“Curse of Dimensionality”**

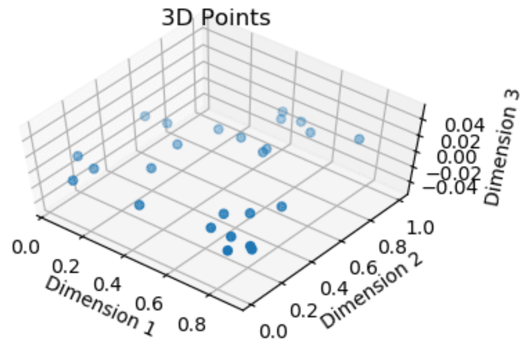


Curse of Dimensionality

- High-dimensional data: datasets with a large number of features, attributes, or dimensions.
- As the number of dimensions increases, the data points become more sparsely distributed throughout the space.
- Sparsity makes it difficult to identify patterns or relationships between data points, as there are fewer data points in a given volume.







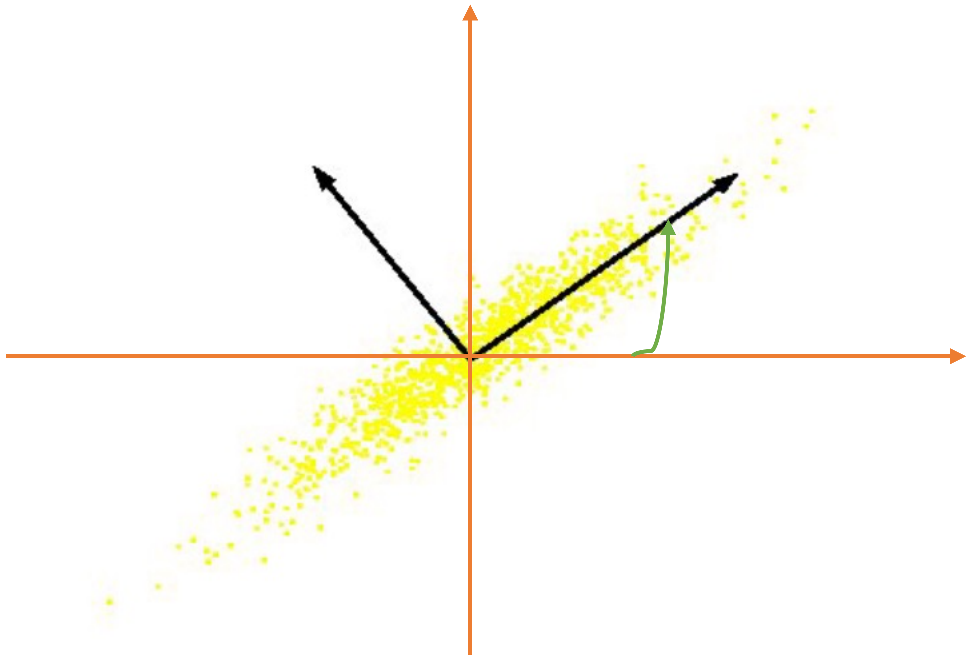
- Solution: Dimension Reduction

Source: https://aiaspirant.com/curse-of-dimensionality/

Dimension Reduction

Principal Component Analysis (PCA)

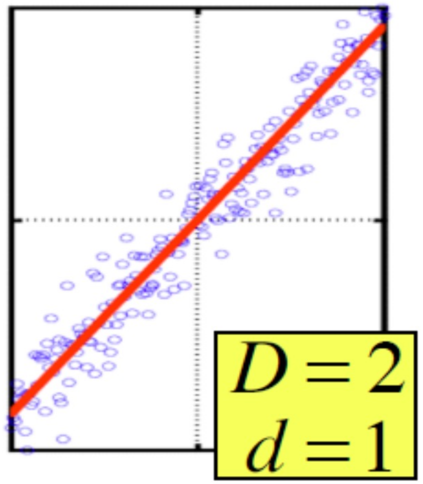
- Unsupervised technique for **extracting variance structure** from high dimensional datasets
- Orthogonal projection or transformation of the data into a (possibly lower dimensional) subspace so that the variance of the projected data is maximized.

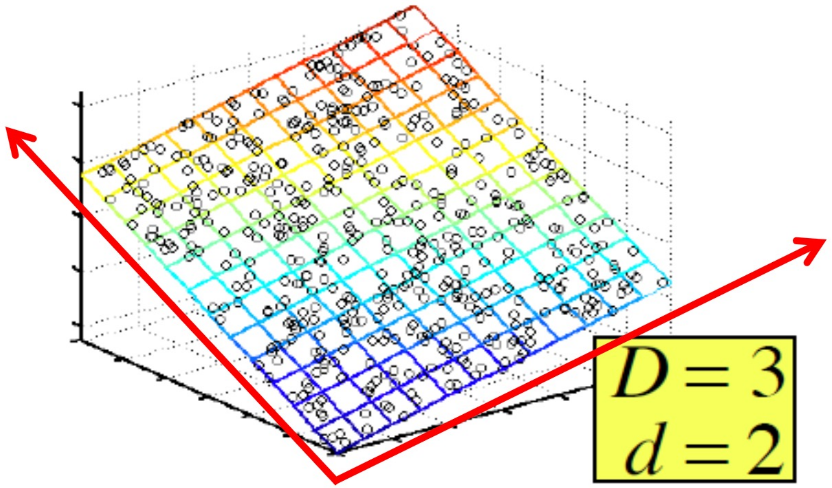


Dimension Reduction

Principal Component Analysis (PCA)

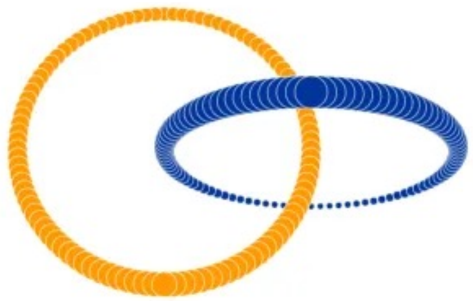
- In case where data lies on or near a low d--dimensional linear subspace, axes of this subspace are an effective representation of the data.
- Identifying the axes is known as Principal Components Analysis, and can be obtained by using classic matrix computation tools (Eigen or Singular Value Decomposition).





Dimension Reduction

t-distributed Stochastic Neighbor Embedding (t-SNE)



- PCA is not suitable for data that are nonlinear relationship
- SNE computes pair-wise similarities
- Pair-wise similarities should stay the same in different dimensions

Step 1: Constructs a probability distribution on pairs in higher dimensions such that similar objects are assigned a higher probability and dissimilar objects are assigned lower probability.

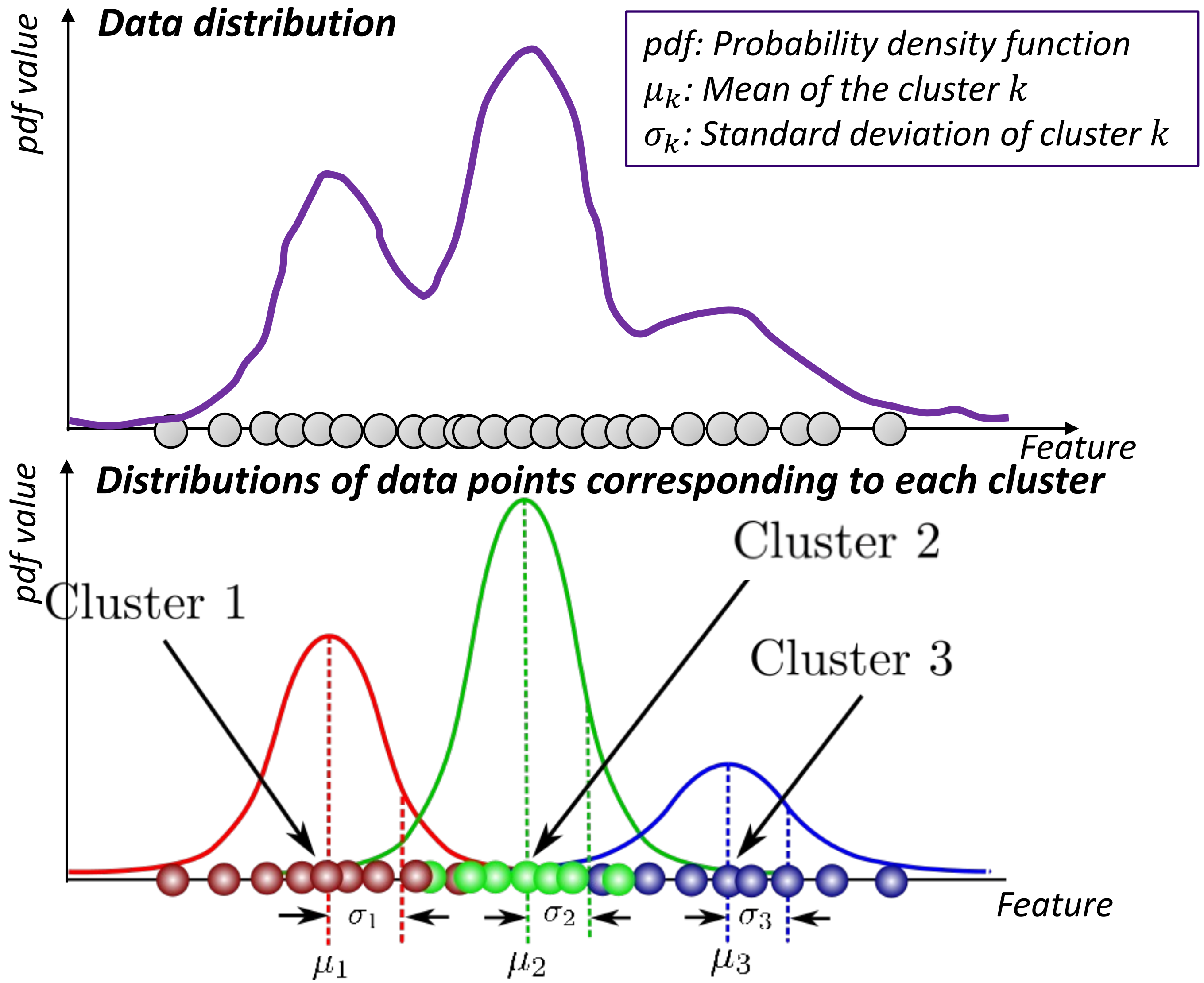
Step 2: t-SNE replicates the same probability distribution on lower dimensions iteratively till the Kullback-Leibler divergence is minimized.

Gaussian Mixture Models

- Probability distribution-based algorithm which assumes data points in each cluster $k = 1, 2, \ldots, K$ arise from corresponding k different Gaussian distributions

- **Goal**: Group data points belong to each Gaussian distribution k together

- Computes the probability of each data point belonging to each of the k distributions (Soft clustering)



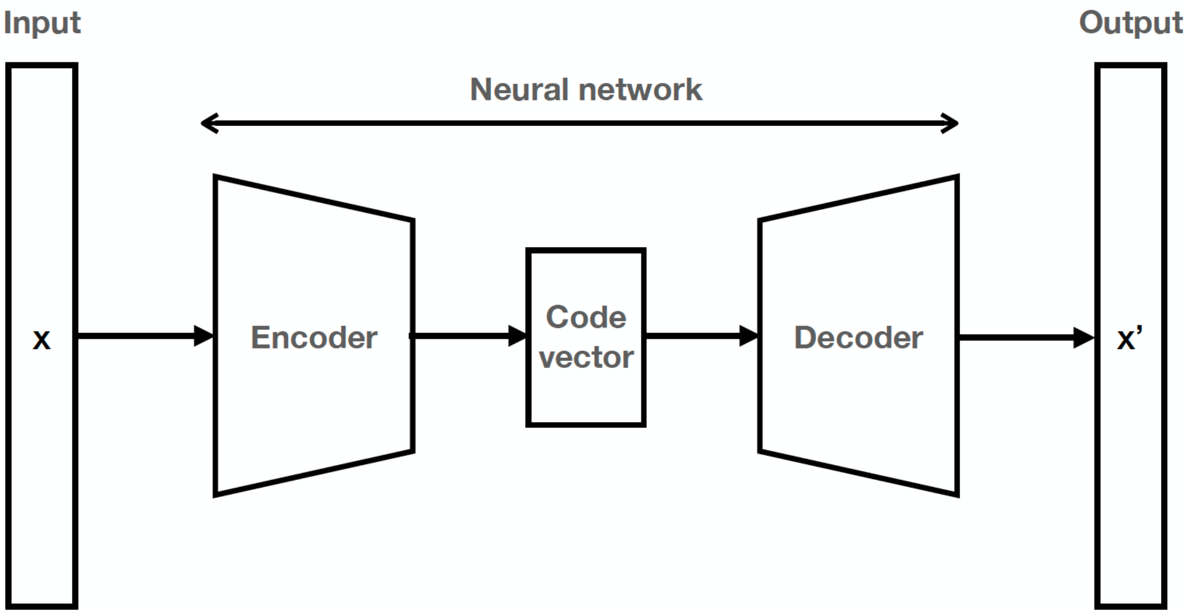
https://towardsdatascience.com/gaussian-mixture-models-explained-6986aaf5a95

Autoencoders

Brief Introduction to Autoencoders

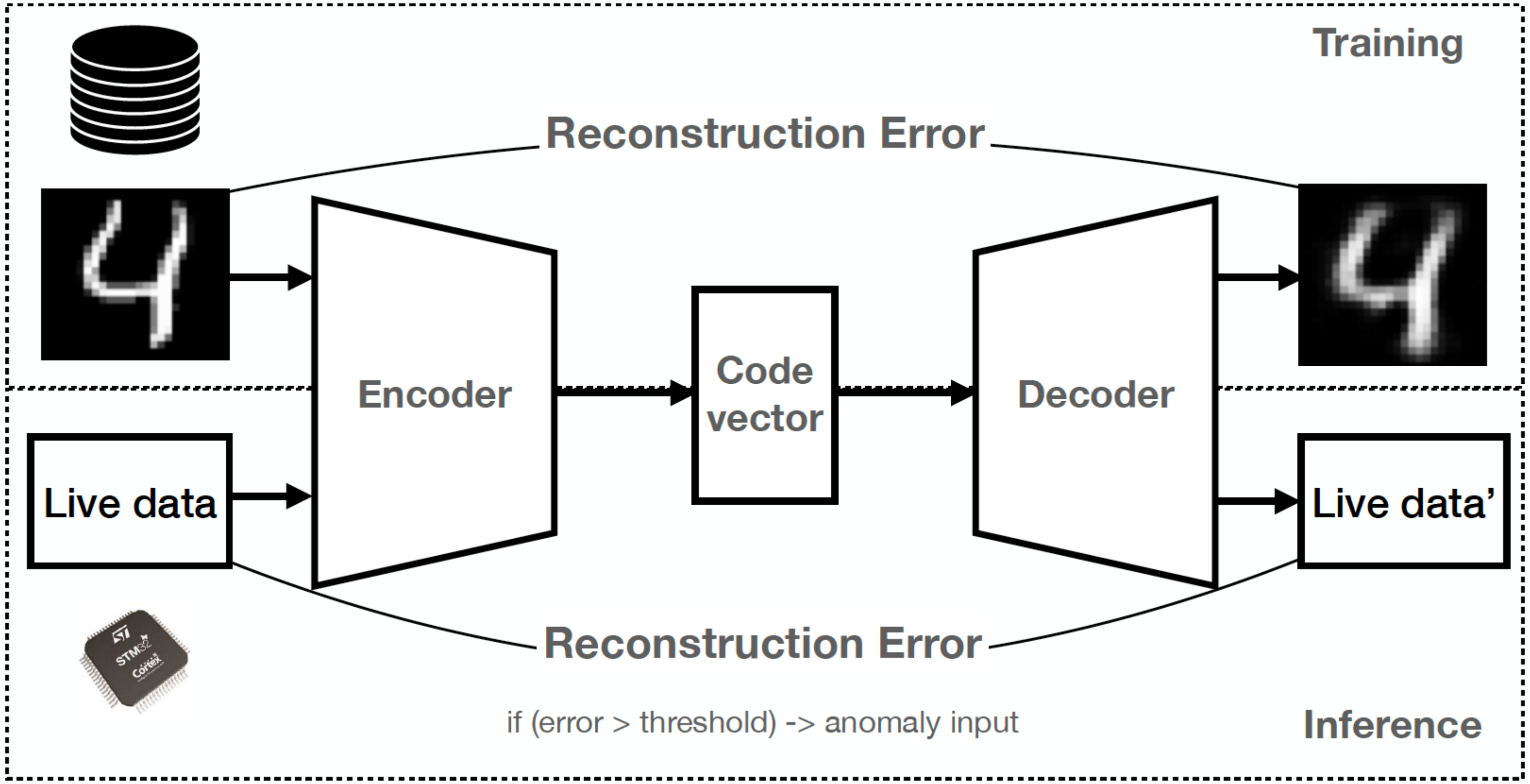
An autoencoder is a neural network that predicts its **input** (ideally x’ = x)

- Encoder: Compress the input into a lower-dimensional code vector
- Code vector: Abstraction of the input
- Decoder: Reconstruct the output from the code vector



Autoencoders

Training : Minimizing the reconstruction error
Inference: Detect anomaly inputs with live data



Autoencoders

Properties of Autoencoders

Unsupervised: We don’t need labels for training.

Data-specific: They can only meaningfully compress data **similar to the training dataset**.

Lossy: The output will not be the same as the input.

In life they say—You don’t know what you don’t know.

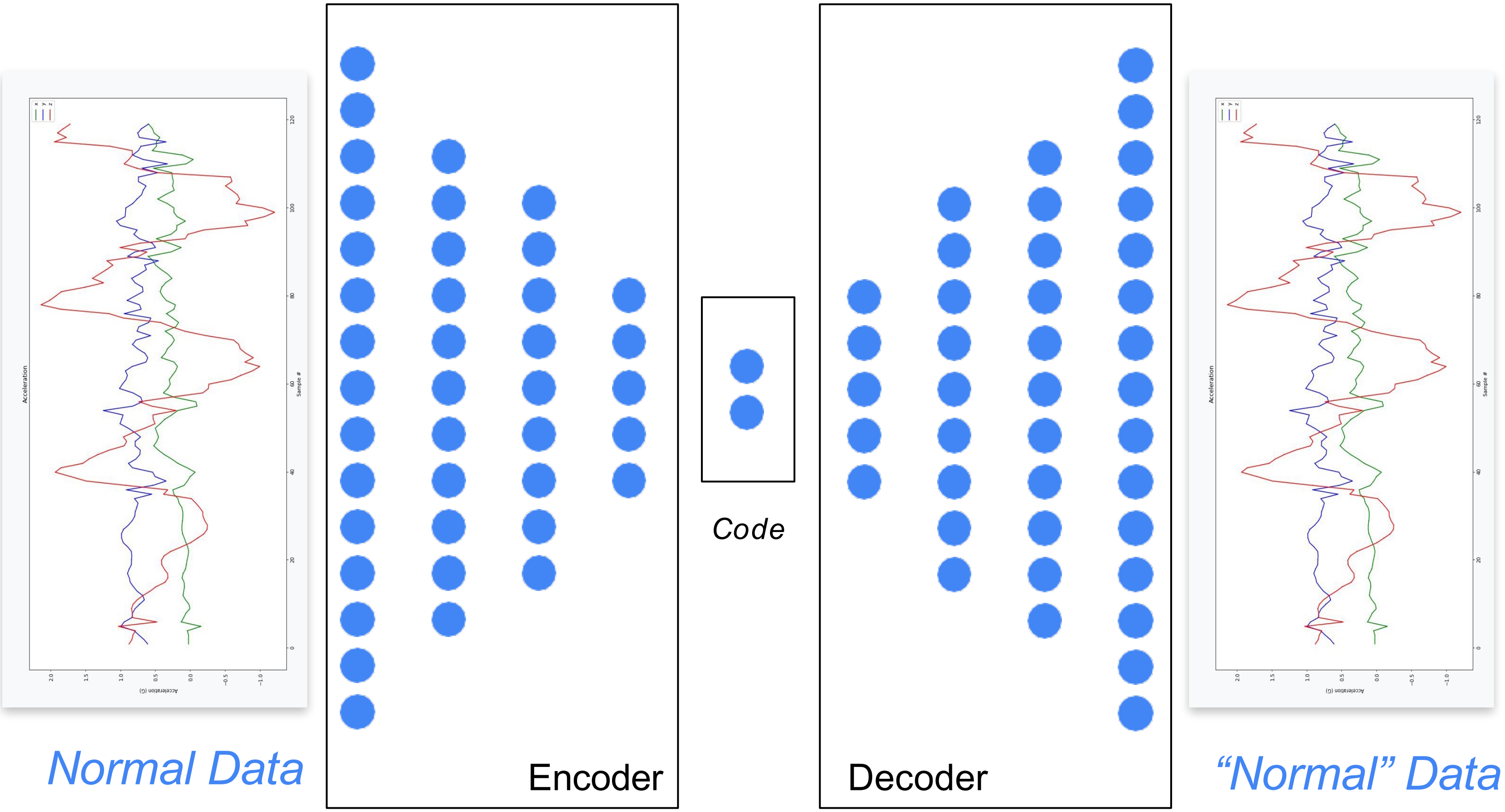
And But you do know what you know!

In TinyML—You know that you don’t know what you think you know!

Until you make TinyML work **You know ⇔ You don’t know**

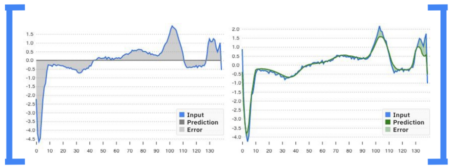
Autoencoders

No anomaly



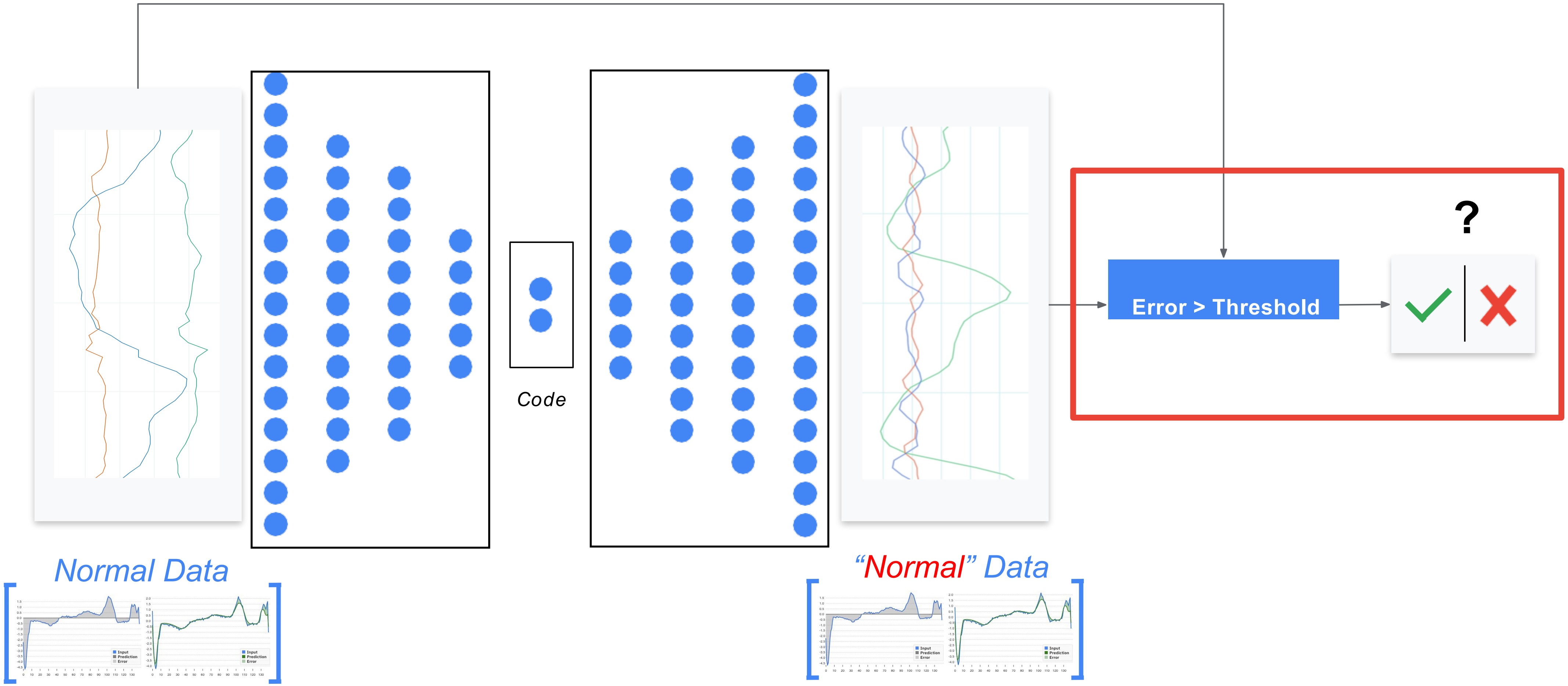
Normal Data

"Normal" Data

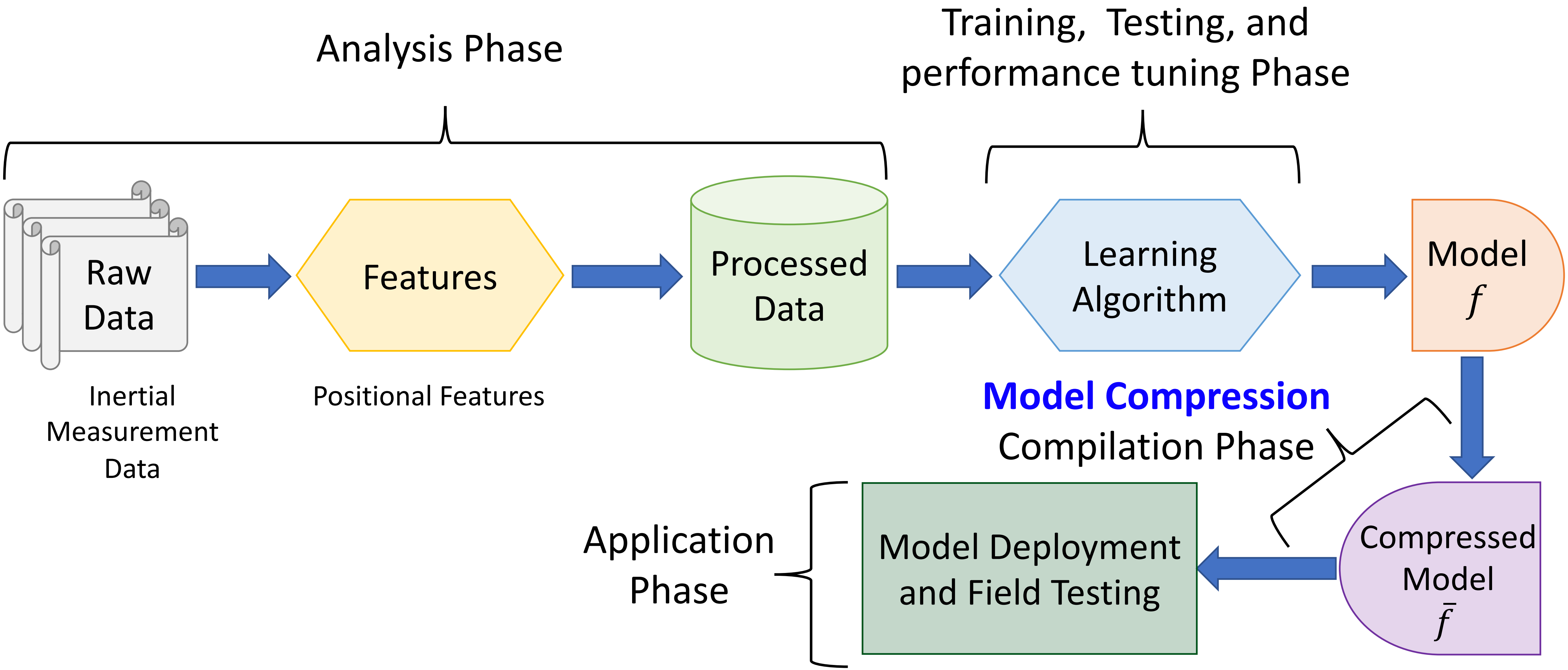


Autoencoders

Anomaly Detected!



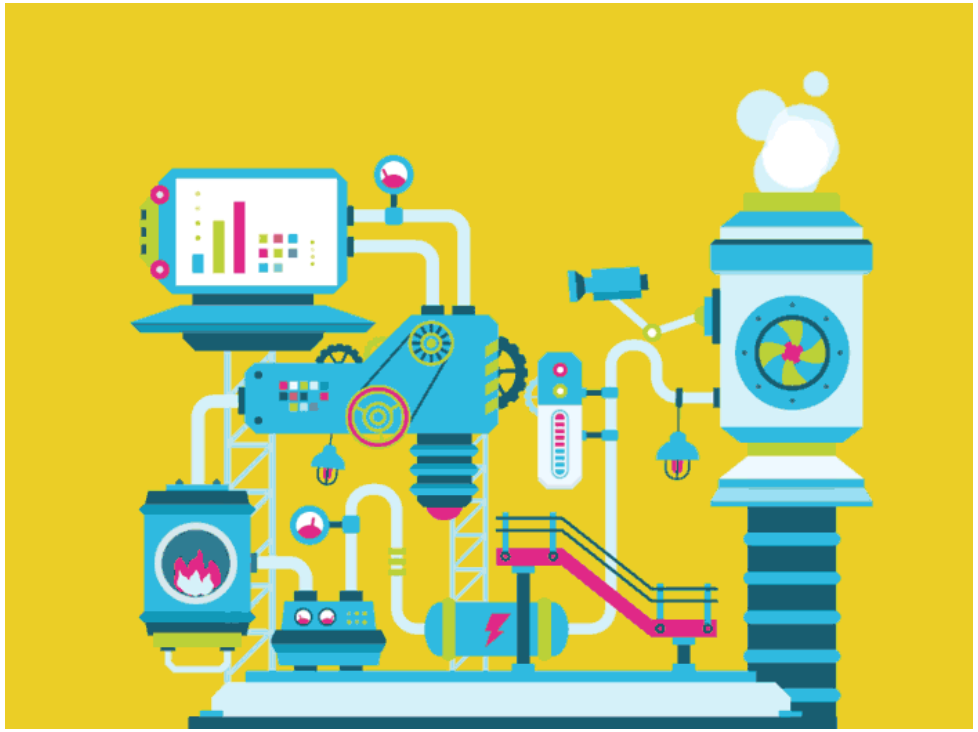
A Schematic View of TinyML and Its Phases



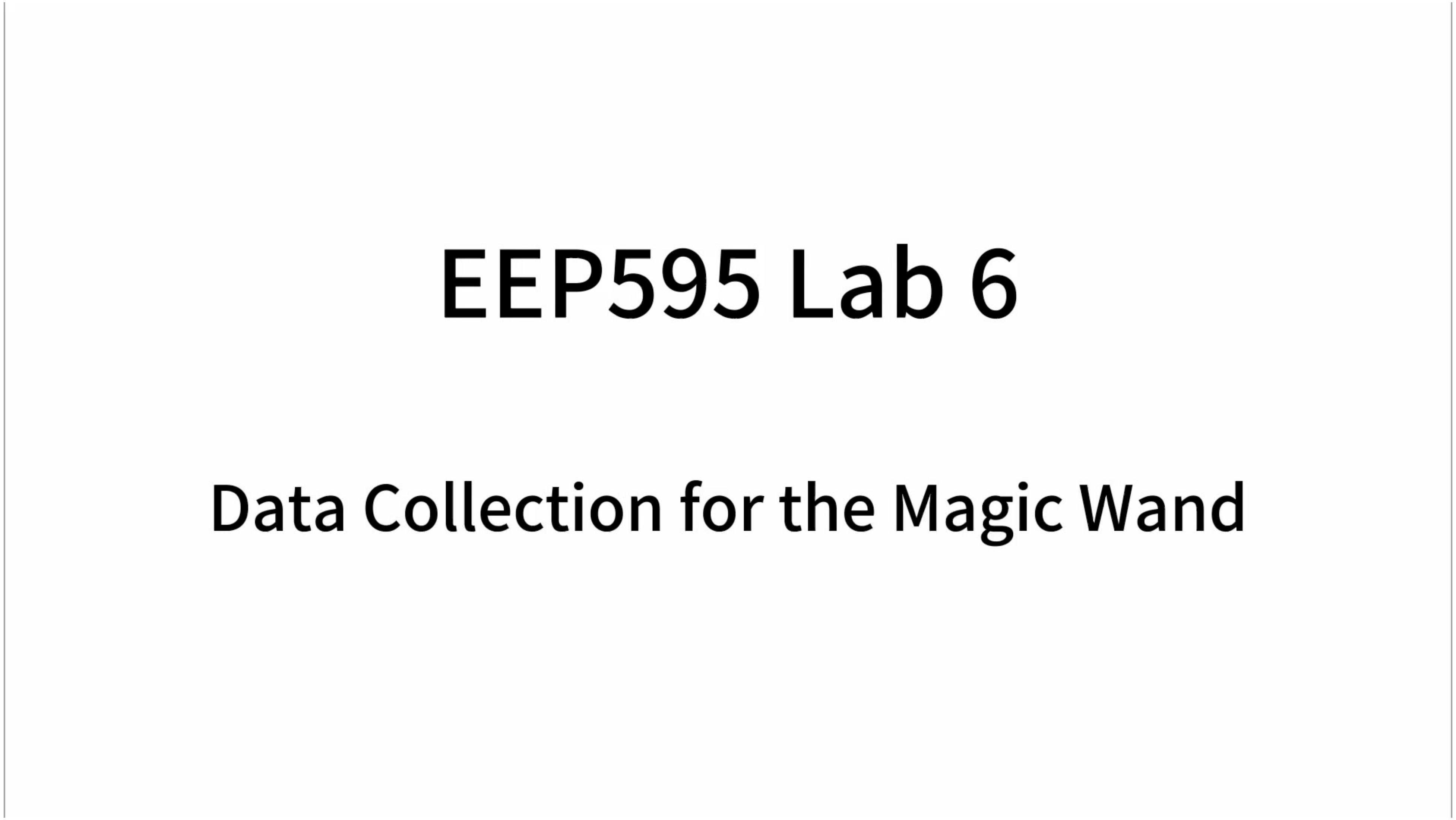

5 min Break

Today's Lab

- All hands on deck for Class 6 - Magic Wand Data Collection
- Both Software and Hardware Lab!
- Testing anomaly detection in fans using Arduino Nano BLE 33 Sense
- Perform k-means clustering for detecting outliers
- Implement t-SNE embedding
- Compress model for TinyML and detect anomalies occurring during running of fan



Data Collection Instructions for Lab 6



Lab 5 - Software - Road Map

1.Examine and understand the data

2.Build a k-means clustering algorithm for sample data

3. Identify and classify anomalies using distance-based detection

4. Perform dimensionality reduction using t-SNE embedding algorithm

5. Load a pre-trained model for fan anomaly detection and perform model compression

6.Evaluate model

Lab 5 - Hardware - Road Map

1. Connect Arduino Boards to your computer

2. Load fan anomaly detection pre-built and compressed model

3. Upload codes to the board

4. Use Serial monitor to identify anomalies while testing with fan

Lab 5 – k-means Clustering

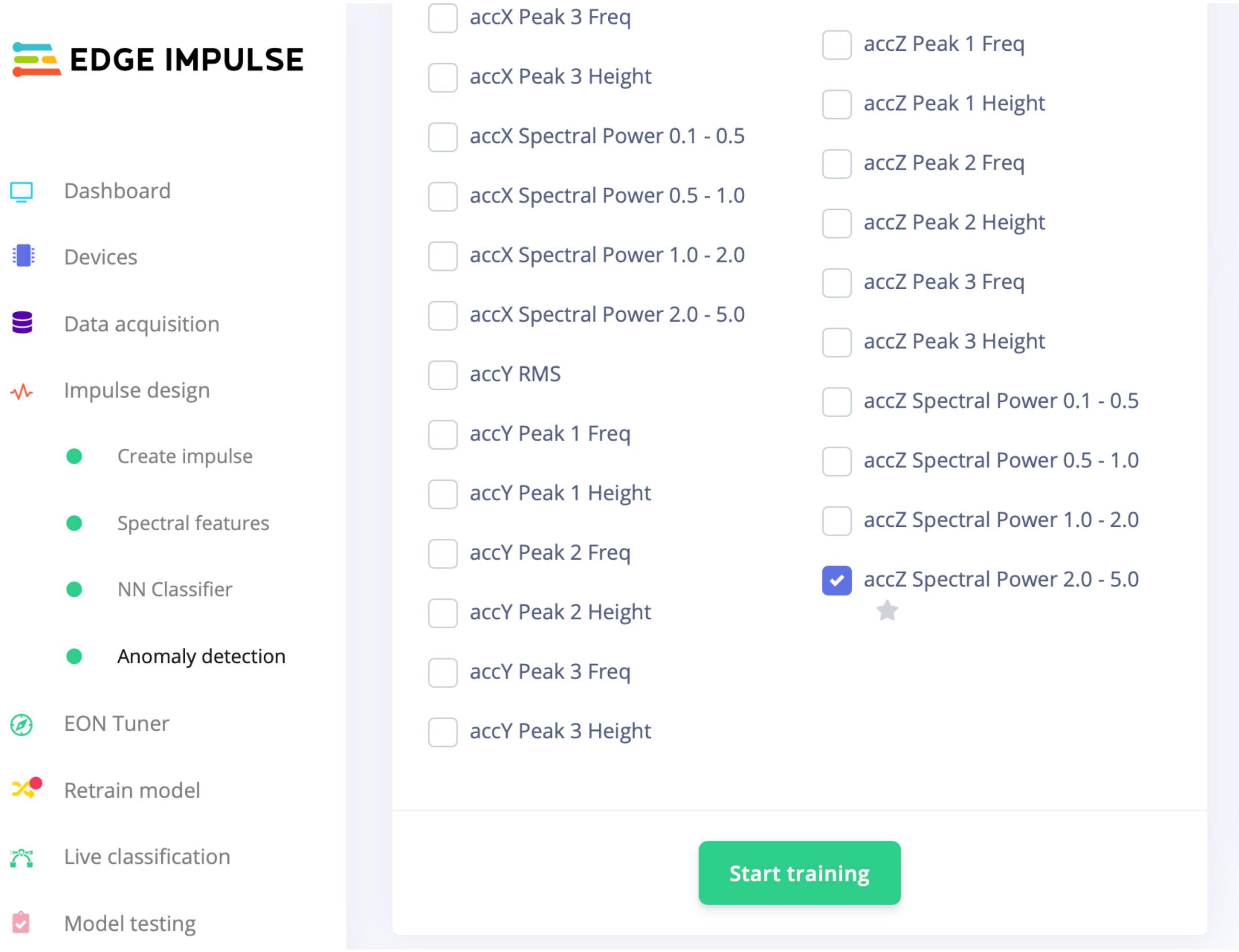
1. Now open EEP595-TinyML-Lab5 (Software).ipynb in Colab

Let's try k-means clustering and t-SNE embedding!

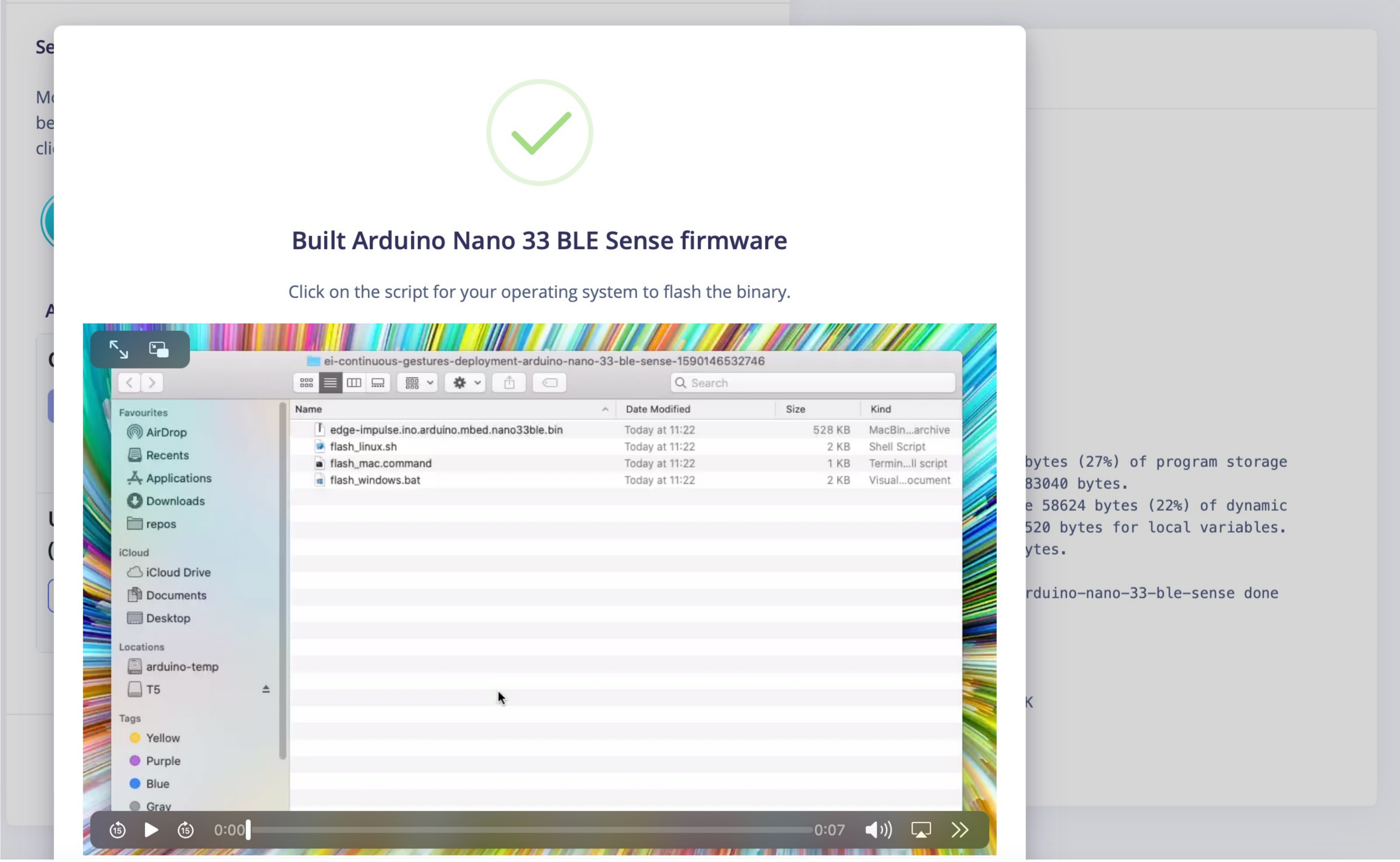
Lab 5 – Software Lab – Edge Impulse

- Open Edge Impulse Fan Monitoring example from: https://studio.edgeimpulse.com/public/47996/latest
- Click on 'Clone this Project'.
- Register for an account on Edge Impulse using your UW ID.
- On the left pane, 'Impulse Design' -> 'Anomaly Detection'.
- Start training the model.
- After training is completed, click on 'Deployment' on the left pane.
- Choose Arduino library -> Arduino Nano 33 BLE Sense -> Build.
- Download the script for flashing on the hardware.

Lab 5 – Model Training



Lab 5 - Optimization



Lab 5 – Hardware Lab

- Open Arduino IDE.
- Connect the BLE 33 board to your laptop and choose the board on the IDE.
- Upload and compile the code onto the board.
- Obstruct / tilt the fan and observe the anomaly values on the Serial Monitor.

Project Environment – Edge Impulse

- Edge Impulse is a great environment for building and running TinyML models.
- It has multiple data acquisition and upload options, support for different hardware devices and their respective IDEs.