Anomaly Detection

Unit 6

Introduction

Anomaly detection is a critical task in machine learning that involves identifying data points, observations, or patterns that do not conform to expected behavior.

These anomalies can be indicative of errors, unusual events, or even malicious activities like fraud.

Types of anamolies

Point Anamolies:

These are individual data points that are significantly different from other data points. They are often referred to as outliers.

Example: In a dataset of daily temperatures, a single day with a temperature of 40°C when all other days are between 20°C and 30°C would be a point anomaly.

Techniques like Local Outlier Factor (LOF), Isolation Forest, and statistical methods (e.g., Z-score) are effective for identifying point anomalies

Types of anamolies

Contextual Anamolies:

These anomalies occur when a data point is anomalous in a specific context or condition but not otherwise. For example, a high temperature might be normal during summer but anomalous during winter.

Example: A transaction of $1000 might be normal for a business account but anomalous for a personal account.

Contextual anomalies require understanding the context in which data is collected. Techniques like conditional probability models or decision trees can be used to identify these anomalies

Types of anamolies

Collective Anamolies:

These are groups of data points that together are anomalous, even if each individual point is not. They often represent a pattern or sequence that is unusual.

Example: In network traffic analysis, a series of seemingly normal packets sent in rapid succession might collectively indicate a denial-of-service attack.

Techniques like clustering, graph-based methods, or sequence analysis (e.g., using Long Short-Term Memory (LSTM) networks) are useful for identifying collective anomalies

Application of Anamoly Detection

Fraud Detection: Identifying unusual transactions in financial systems.

Network Security: Detecting malicious activity or intrusions.

Healthcare: Identifying unusual patterns in patient data that may indicate health issues.

Quality Control: Identifying defects or anomalies in manufacturing processes.

Techniques for Anomaly Detection

Statistical methods for anomaly detection rely on identifying data points that deviate from expected statistical distributions or patterns. These methods are often simple to implement and can be effective when the dataset is small or when the data follows a specific statistical distribution.

Percentile Method: Identifies data points that fall outside a specific percentile range.

Interquartile Range (IQR) Method: Uses the difference between the 75th percentile (Q3) and the 25th percentile (Q1) to define a range for normal data; points outside this range are considered anomalies.

Z-Score Method: Calculates how many standard deviations away from the mean a data point is; points with a high Z-score are considered anomalies.

Techniques for Anomaly Detection

Distance-based methods identify anomalies by measuring the distance between data points. These methods are effective when the data is sparse or when anomalies are far from other data points.

K-Nearest Neighbors (KNN): Identifies anomalies based on the distance to their nearest neighbors; points with a large distance are considered anomalies.

Techniques for Anomaly Detection

Density-based methods detect anomalies by identifying regions of low density in the data. These methods are particularly effective for identifying local anomalies.

Local Outlier Factor (LOF): Measures the local density of a point compared to its neighbors to identify outliers.

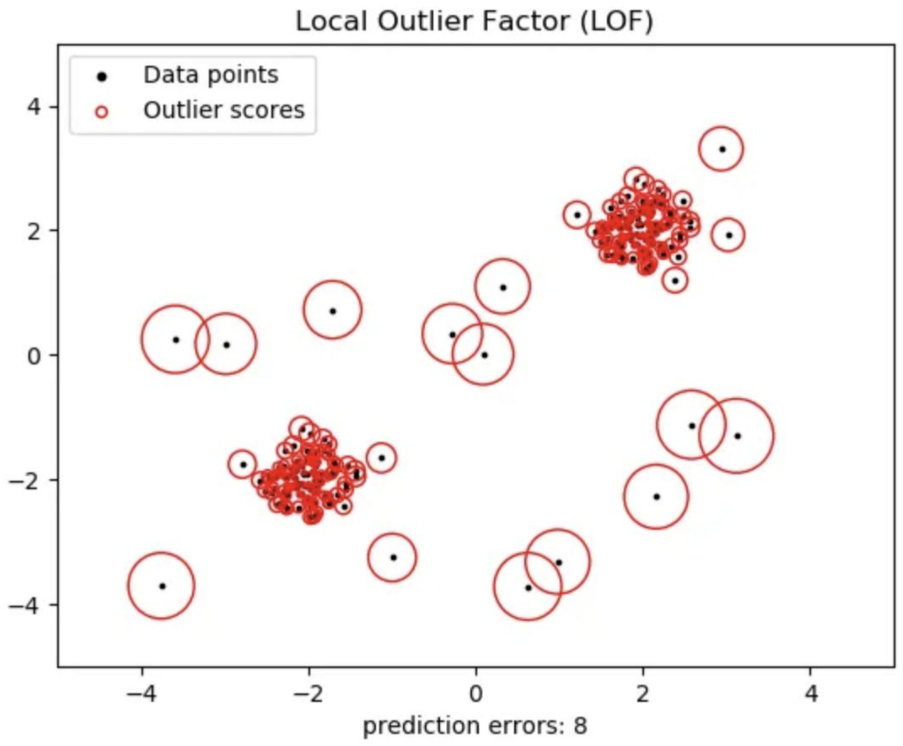
Isolation Forest: Uses decision trees to isolate anomalies by splitting data points randomly until anomalies are separated from the rest.

Local Outlier Factor (LOF)

LOF is the most well-known and widely used local anomaly detection algorithm.

It carries the idea of nearest neighbors to determine the anomaly or outlier score.

The Local Outlier Factor (LOF) algorithm is an unsupervised anomaly detection method which computes the local density deviation of a given data point with respect to its neighbors. It considers as outliers the samples that have a substantially lower density than their neighbors.



Isolation Forest

Isolation Forest is an unsupervised machine learning algorithm for anomaly detection.

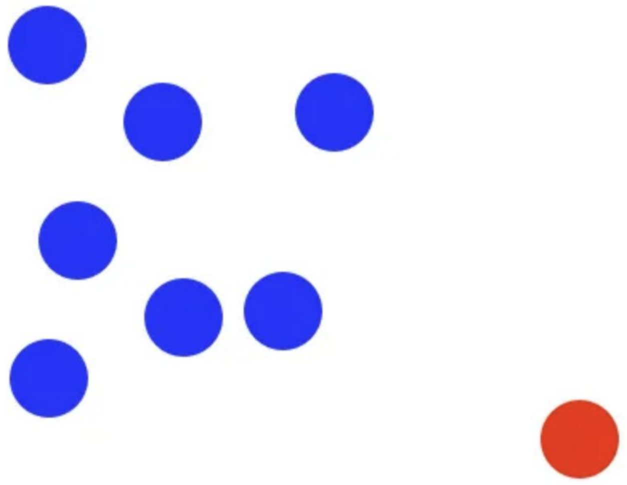
As the name implies, Isolation Forest is an ensemble method (similar to random forest).

In other words, it use the average of the predictions by several decision trees when assigning the final anomaly score to a given data point.

Unlike other anomaly detection algorithms, which first define what's "normal" and then report anything else as anomalous, Isolation Forest attempts to isolate anomalous data points from the get go

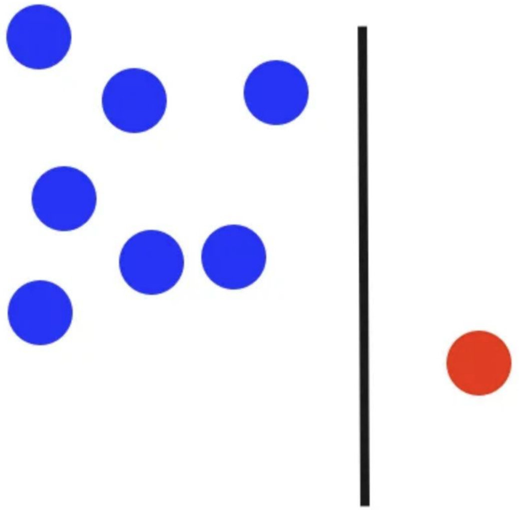
Example

Suppose we have the data points



Example

The isolation forest algorithm selects a random dimension (in this case, the dimension associated with the x axis) and randomly splits the data along that dimension.

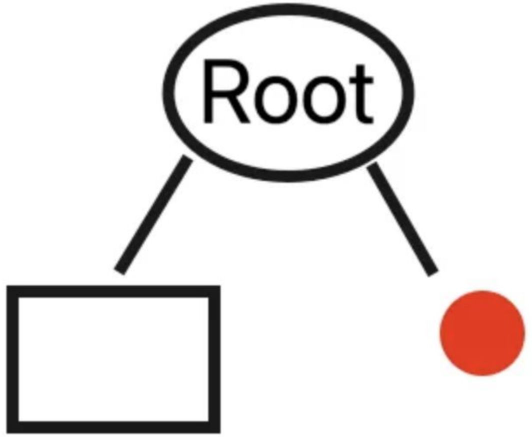


Example

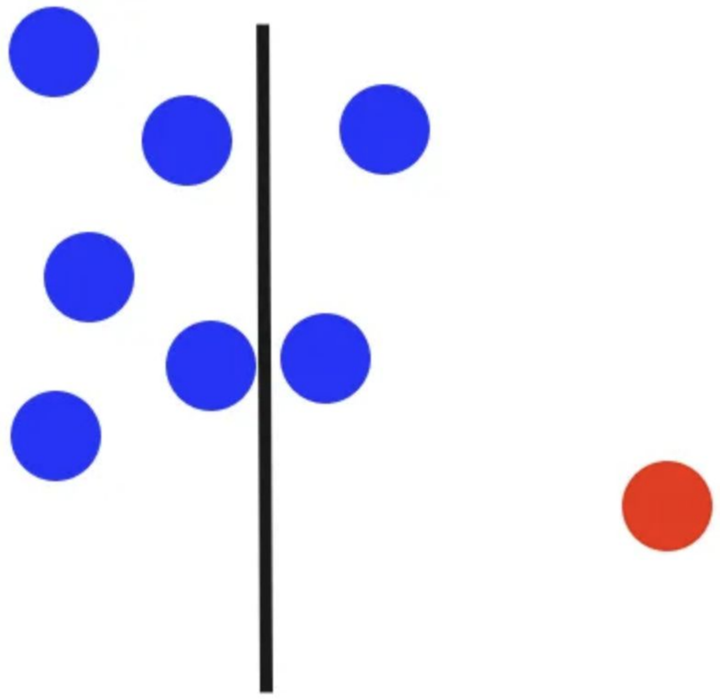
The two resulting subspaces define their own sub tree.

In this example, the cut happens to separate a lone point from the remainder of the dataset.

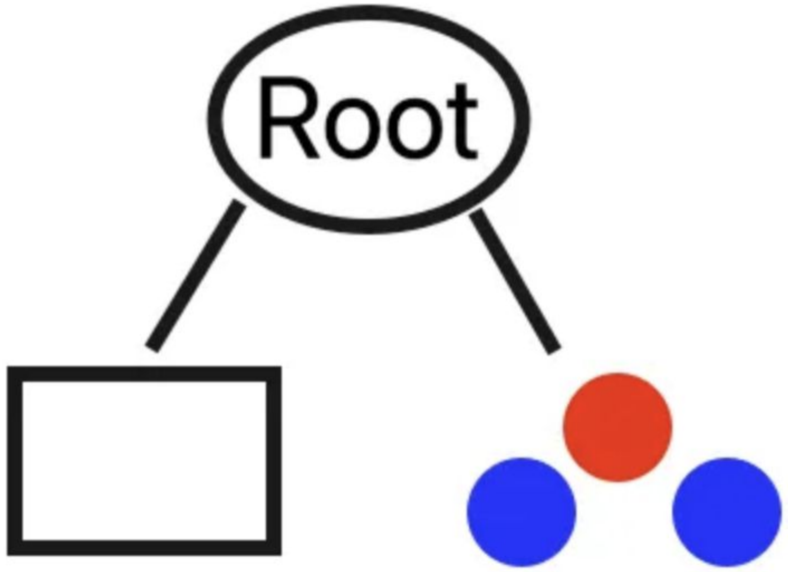
The first level of the resulting binary tree consists of two nodes, one which will consist of the subtree of points to the left of the initial cut and the other representing the single point on the right.



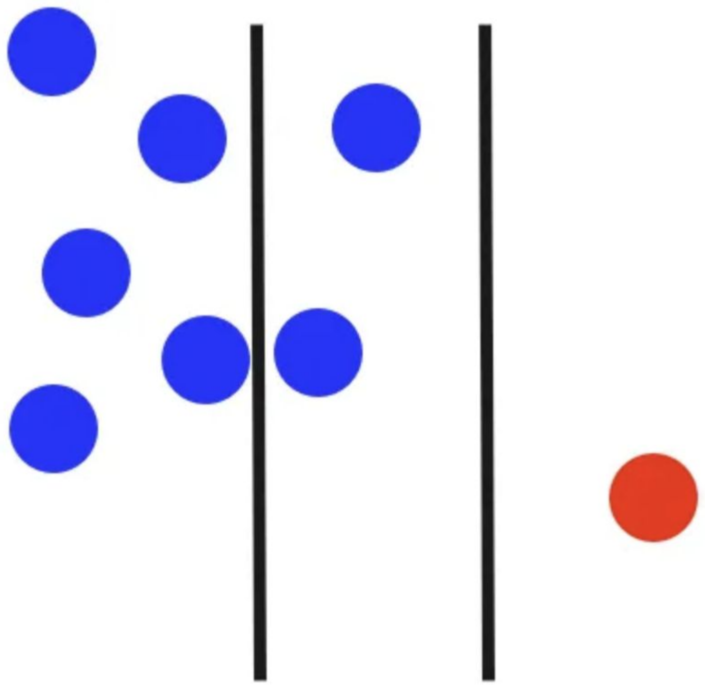
It's important to note, the other trees in the ensemble will select different starting splits. In the following example, the first split doesn't isolate the outlier.



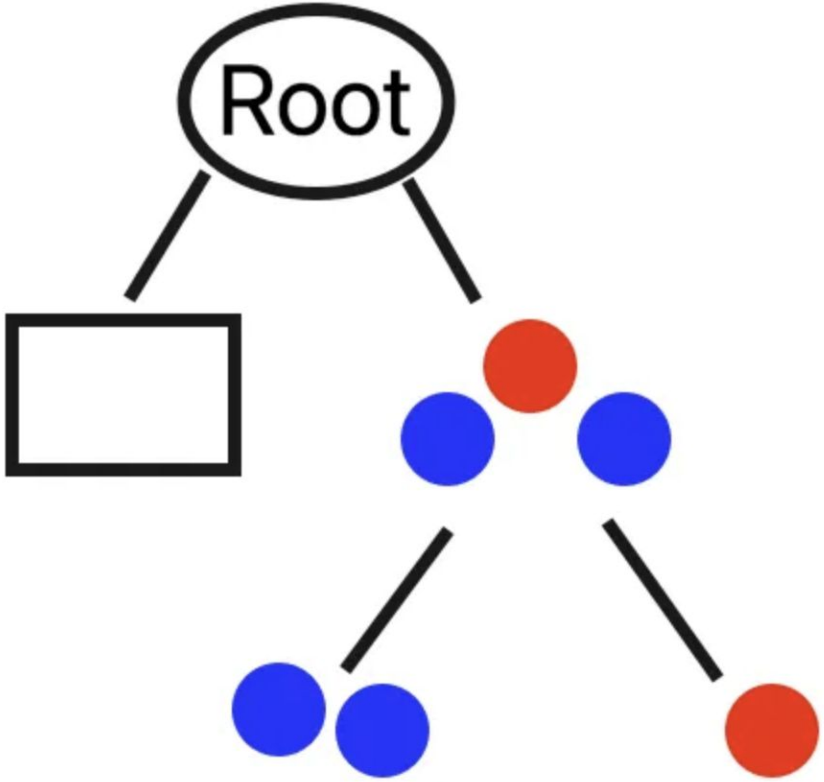
We end up with a tree consisting of two nodes, one that contains the points to the left of the line and the other representing the points on the right side of the line.



The process is repeated until every leaf of the tree represents a single data point from the dataset. In our example, the second iteration manages to isolate the outlier.



After this step, the tree would look as follows:



Techniques for Anomaly Detection

Clustering-based methods group similar data points into clusters and identify anomalies as points that do not fit well into any cluster.

K-Means Clustering: Groups data points into clusters based on similarity; points far from cluster centers are considered anomalies.

Hierarchical Clustering: Builds a hierarchy of clusters and identifies anomalies based on their position in the hierarchy.

Techniques for Anomaly Detection

One-Class Classification: Trains a model on normal data to classify new data points as either normal or anomalous.

Isolation Forest: Uses decision trees to isolate anomalies by splitting data points randomly until anomalies are separated from the rest.

One class SVM

The One-Class Support Vector Machine (SVM) is a variant of the traditional SVM. It is specifically tailored to detect anomalies.

Its primary aim is to locate instances that notably deviate from the standard.

Unlike conventional Machine Learning models focused on binary or multiclass classification, the one-class SVM specializes in outlier detection within datasets.

One class SVM

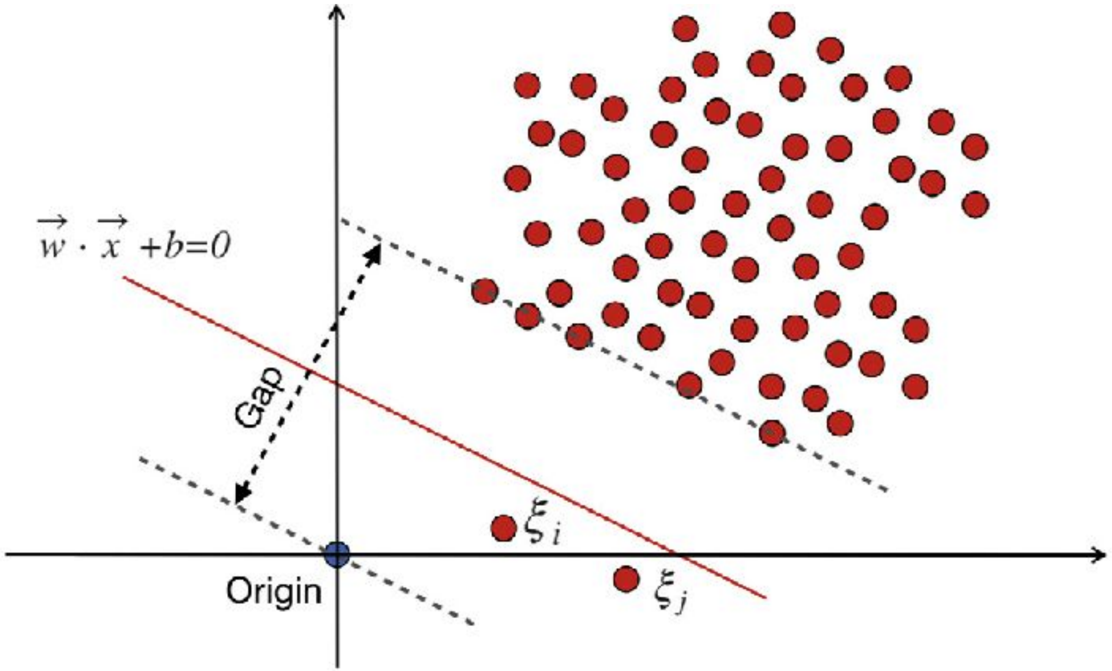
One-class SVMs represent a variant of the traditional SVM algorithm primarily employed for outlier and novelty detection tasks.

Unlike traditional SVMs, which handle binary classification tasks, One-Class SVM exclusively trains on data points from a single class, known as the target class.

One-class SVM aims to learn a boundary or decision function that encapsulates the target class in feature space, effectively modeling the normal behavior of the data

One-class SVM aims to discover a hyperplane with maximum margin within the feature space by separating the mapped data from the origin.

On a dataset Dn = {x1, . . . , xn} with xi ∈ X (xi is a feature) and n dimensions



Kernel functions in One class Classification

Kernel functions play a crucial role in One-Class SVM by allowing the algorithm to operate in higher-dimensional feature spaces without explicitly computing the transformations.

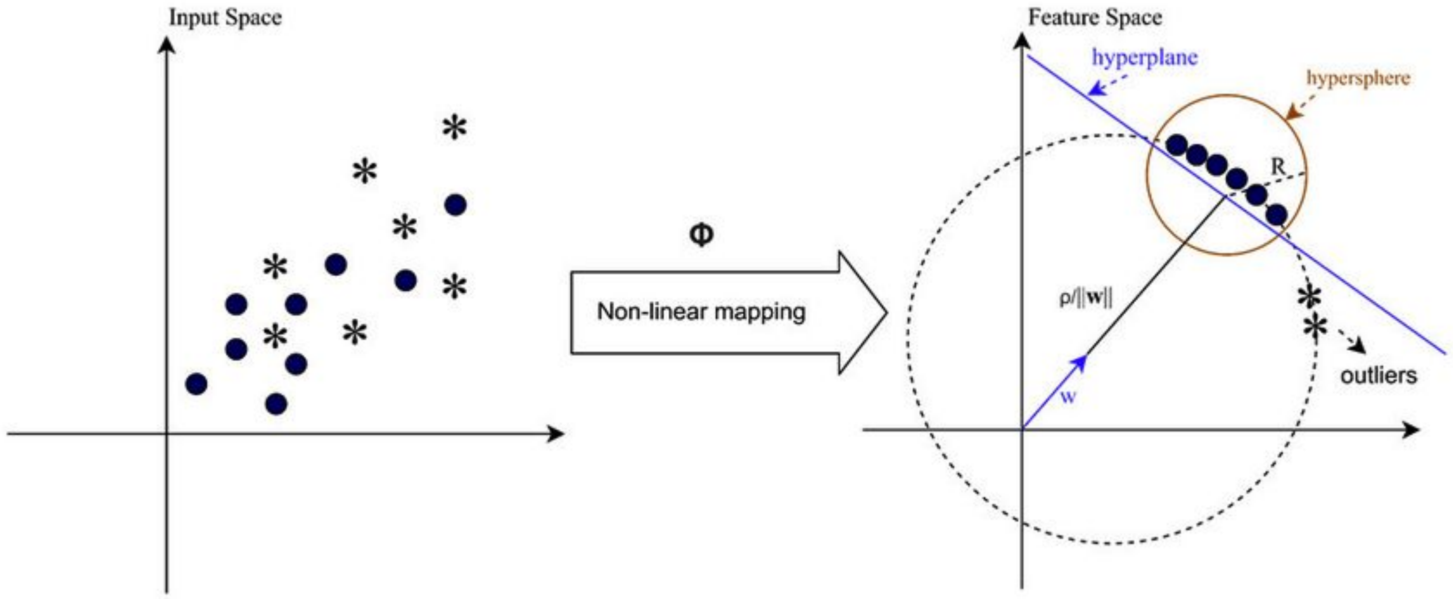
In One-Class SVM, as in traditional SVMs, kernel functions are used to measure the similarity between pairs of data points in the input space.

Common kernel functions used in One-Class SVM include Gaussian (RBF), polynomial, and sigmoid kernels.

These kernels map the original input space into a higher-dimensional space, where data points become linearly separable or exhibit more distinct patterns, facilitating learning.

Kernel functions in One class Classification

By choosing an appropriate kernel function and tuning its parameters, One-Class SVM can effectively capture complex relationships and non-linear structures in the data, improving its ability to detect anomalies or outliers.



Anomaly Detection in high dimension

Anomaly detection in high-dimensional data involves identifying observations that significantly deviate from the majority of the data.

This task is challenging due to the "curse of dimensionality," which can lead to increased computational complexity and reduced model performance.

One of the approach is to first reduce the dimensionality using PCA or LDA and then detect the anomaly using the above approaches

Anomaly Detection in high dimension

Autoencoders are powerful tools for handling high-dimensional datasets by reducing their dimensionality while preserving essential information.

Autoencoders achieve dimensionality reduction by using an encoder to map the high-dimensional input data into a lower-dimensional latent space.

The decoder then reconstructs the original data from this latent space, ensuring that the most important features are retained

Some of the different types of autoencoders for high dimension

Undercomplete Autoencoders:

These are the simplest form of autoencoders and are specifically designed for dimensionality reduction by having a bottleneck layer with fewer neurons than the input layer.

They are effective for reducing high-dimensional data into a lower-dimensional representation while retaining essential features.

Sparse Autoencoders:

These autoencoders add a sparsity constraint to the hidden layer, ensuring that only a subset of neurons are active at any given time.

This helps in preventing overfitting and is particularly useful for high-dimensional data where feature selection is crucial.

Some of the different types of autoencoders for high dimension

Denoising Autoencoders:

These autoencoders are trained on noisy data and learn to reconstruct the original data from the noisy version.

They are robust against noisy high-dimensional data and can effectively learn meaningful representations.

Some of the different types of autoencoders for high dimension

Convolutional Autoencoders (CAE):

These are particularly useful for image data, which is inherently high-dimensional.

CAEs use convolutional layers for encoding and decoding, making them efficient for spatial data like images.

Variational Autoencoders (VAE):

VAEs are probabilistic models that learn a continuous and structured representation of data.

They are effective for high-dimensional data by providing a generative model that can sample new data points similar to the training data