Q1: What are the pros and cons of anomaly detection and misuse detection?

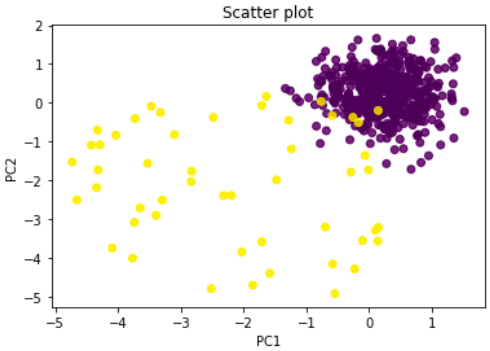
Ans:

	Anomaly Detection	Misuse Detection
Definition	Normal behavior is defined first, and all other is defined as abnormal.	Abnormal behavior is defined first, and all other is defined as normal.
Pros	- Can detect new and unseen attacks	- Simplicity of adding known attacks to the model
Cons	- How to be defined "Normal", identifying normal activity vs. truly abnormal is extremely challenging	- Need to be equipped with a well-defined set of attack signatures populated in their database completely and sufficiency - Cannot detect the attack as the attack signature is not present in the existing database

Q2: Describe how to use an autoencoder to detect an anomaly? Probably you need draw a graph.

An autoencoder is a type of ANN used to learn efficient data coding in an unsupervised manner. The aim of an autoencoder is to learn a representation for a set of data, typically for dimensionality reduction. Different from PCA is doing non-linear interactions between the variables. How we can exploit that is by utilizing a loss distribution of rebuilt inputs to outputs (which turns out to be Gaussian) and making the assumption that any outliers will be anomalies since they faulter well outside the parameters of what the model considers "within the expected distribution". Therefore the trained Autoencoder can reconstruct normal samples, but cannot reconstruct anomaly samples.

Q3: Describe how to use PCA to detect an anomaly? Probably you need draw a graph.



Left illustration description: Purple points are normal, yellow points are anomaly.

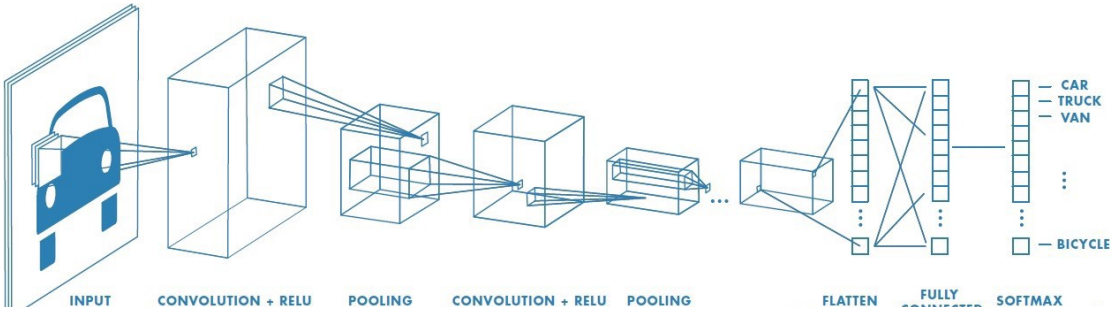
PCA performs a linear mapping of the data to a lower-dimensional space in such a way that the variance of the data in the low-dimensional representation is maximized. Meanwhile, according to PCA math

calculation, PCA is sensitive to outlier. After the points projected to new vector space, we can calculate Mahalanobis distance to estimate anomaly score to determine whether the point is outlier.

Q4: Using one-hot encoding to encode an article (having lots of words) is impractical. What else method can you adopt?

One-hot encoding is very traditional method. It will cause “curse of dimensionality” and there’s strong assumption that every word is independent. Currently we can use word embedding, it considers the correlation and order of every word and uses ANN architecture to project to a new vector space which is small dimension compared to one-hot encoding. For example, there’s Word2Vec and Glove to implement it. The state-of-the-art method is bidirectional sequence to sequence model, like ELMO, BERT ...and so on.

Q5: For an CNN (shown below, note that the second last layer is ”FULLY CONNECTED”), please describe (1) what is the purpose of first part layers (from input to FLATTEN), and (2) what is the purpose of the last few layers (FLATTEN to SOFTMAX).



(1) Fully connected input layer (Flatten)

This layer takes the output of the previous layers, “flattens” them and turns them into a single vector that can be an input for the next stage.

(2) Fully connected layer

This layer takes the inputs from the feature analysis and applies weights to predict the correct label and go through SoftMax to give the final probabilities for each label.

Q6: Tell my why the above “FLATTEN” layer can be viewed as a representation of the original INPUT.

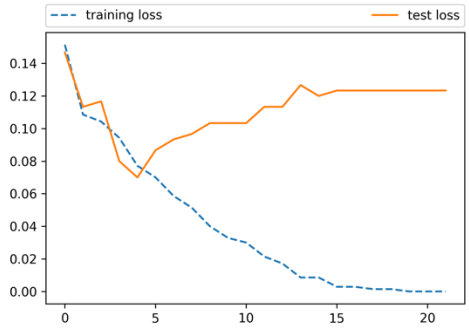
After convolution layer and pooling layer to downsize image, and last, we flatten the output to fit fully connected layer neural network. So “flatten” layer is down dimension of image and filtered trivial features. It will be enough to represent original input.

Q7: Why do we need Min-Max Scaling or Z-score Normalization? How can them help the training process?

(1) Because each feature has its own range of data, for example, height range from 100~220(cm) normally and weight range from 40~110(kg) normally, they are not in the same level. We need to normalize our data is to change the values of numeric columns in the dataset to a common scale, without distorting differences in the ranges of values.

(2) In machine learning viewpoint, if you do not normalize your data, different features have wide ranges of values, it cause that gradients may end up taking a long time and can oscillate back and forth and take a long time before it can finally find its way to the global/local minimum. Therefore, we need to do normalization to take on similar ranges of feature values so that gradient descents can converge more quickly.

Q8: How do you know your model overfit? How do you prevent from overfitting in a neural network?



(1) We need to check our training loss and testing loss, if training loss decreased and testing loss increase, we're careful about the model is overfitting.

(2) When the model is overfitting, we can do "dropout", "L1/L2 regularization", "modify architecture of neural network" … and so on.

Q9: What is the difference between noise and anomaly? How to remove them?

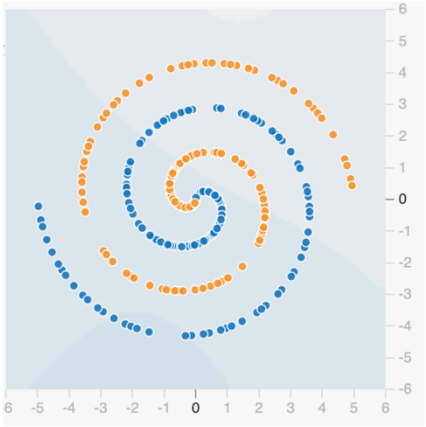
Noise is anything that's not the "true" signal. It may have values close your true signal. An outlier is something that's much different than the other values. The simple way is that we can use statistical method like IQR or Z-score to remove "outliers". PCA is also the way can remove noise.

Q10: What is the kernel of a CNN?

Ans: The kernel of CNN is used to extract the features from the images and is also equal to filter. Kernel moves on the input data by the stride value. It will find the most important proportion of image as features and be enough to represent a part of the

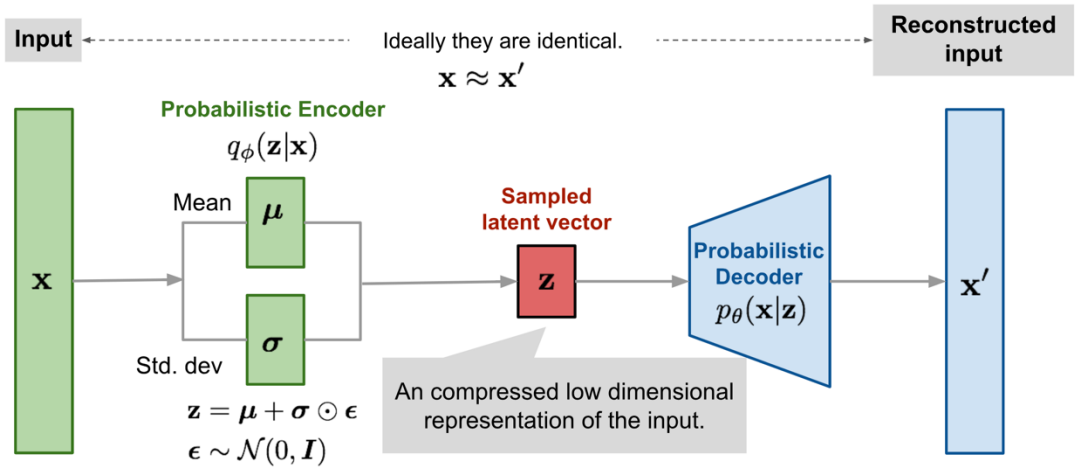
image to recognize.

Q11: Why a neural network can classify orange points and blue points? Give us some high-level explanation.



Because neural network will do 3 things, rotation, translation and distortion. These three moves will let the two categories of points depart from each other and come up a hyperplane to separate two categories of points.

Q12: What is the epsilon in VAE? Why it is a normal distribution N(0, I)?



Decoder randomly samples from true posterior Z~ q(z | ϕ,x). To implement encoder and decoder as a neural network, you need to backpropagate through random sampling and that is the problem because backpropagation cannot flow through random node. To overcome this obstacle, we use reparameterization trick. The epsilon remains as a random variable (sampled from a standard normal distribution) with a very low value thereby not causing the network to shift away too much from the true distribution. It's safe to reiterate here that distribution ϕ (parameterized by the mean and log-variance vectors) is still being learned by the network. This idea actually allowed a VAE to train in an end-to-end manner.