How Linear Algebra and Probability Distributions Concepts Are Utilized in Various ML Models

Carl Lochstampfor Jr

Department of Cybersecurity, Old Dominion University

CYSE 420 — Applied Machine Learning in Cybersecurity

August 29, 2026

How Linear Algebra and Probability Distributions Concepts Are Utilized in Various ML Models

Overview of Key Concepts

Machine learning in cybersecurity depends on two areas of mathematics. Linear algebra provides the structures that hold security data and the operations that process it quickly. Probability and statistics provide the language for expressing how confident a model is that an observation is hostile or malicious. Each concept below is introduced using the kind of data a security model actually receives in the real world.

Linear Algebra Concepts

Vectors and Matrices. A vector is an ordered list of numbers, and in machine learning, it usually represents a single observation. A single network connection can be described by x = [duration, bytes sent, bytes received, packet count, destination port], a five-element vector. Stacking many observations together produces a matrix. For example, a capture of 10,000 connections described by those same five features forms a 10,000 × 5 matrix, where each row represents one connection, and each column represents one feature. This structure matters because once traffic is stored in a matrix, a single operation can score every connection at once rather than looping through them individually, saving precious time and resources.

Tensors. Tensors generalize vectors and matrices to arbitrary dimensions (Goodfellow et al., 2016). A scalar is a rank-0 tensor, a vector is rank 1, and a matrix is rank 2. Security data often needs a third dimension because the order of events matters. For example, a batch of 64 login sessions, each observed over 20 time steps (or stamps), each step described by 8 features, forms a 64 × 20 × 8 tensor. The course slides illustrate the same idea with one-hot-encoded text, where two sentences of five words, drawn from a six-word vocabulary, produce a (2, 5, 6) tensor (Wang, 2026).

Dot Products and Matrix Multiplication. The dot product multiplies the corresponding elements of two vectors and sums the results. Given a learned weight vector w, the suspicion score for one connection is the dot product w · x. Matrix multiplication extends this to whole datasets, so the product Xw produces that score for all 10,000 connections in a single step. Matrix multiplication is associative and distributive but not commutative, meaning AB and BA are not the same, so the order of the operands is fixed by the shapes involved (Wang, 2026).

Norms. A norm measures the size of a vector, and therefore also the distance between two vectors. The L2 norm is the square root of the sum of the squared elements, which is straight-line distance. The L1 norm adds absolute values. Both satisfy non-negativity, positive scalability, and the triangle inequality (Wang, 2026). Norms are useful in security models in two ways. The L2 distance between an observed traffic vector and a learned baseline vector can serve directly as a behavioral anomaly score, and adding the weight norm to the training objective prevents any single feature from dominating the model.

Identity and Inverse Matrices. The identity matrix I leaves any vector unchanged when multiplied, and the inverse of a matrix reverses its transformation, so the system $Ax = b$ can be solved as $x = A^{-1}b$. A scaled identity matrix appears in L2 regularization, where the term λI is added before inversion to penalize large coefficients (Wang, 2026).

Eigen-decomposition. Eigen-decomposition breaks a square matrix into eigenvectors and eigenvalues, defined by $Av = \lambda v$, where multiplying the eigenvector v by A changes only its scale (Wang, 2026). This is the basis of principal component analysis, which identifies the directions along which the data varies most and allows a large feature set to be reduced to a smaller one before further processing.

Probability Distributions

Bernoulli Distribution. The Bernoulli distribution models a single binary outcome and is defined by one parameter, p, the probability of the positive class (Goodfellow et al., 2016). It appears in security classification at two levels. A detector that outputs 0.93 for a connection is describing a Bernoulli distribution with $p = 0.93$ over the labels malicious and benign. Individual features are often Bernoulli as well, such as whether a specific word appears in an email.

Categorical Distribution. The categorical distribution extends the Bernoulli case to k mutually exclusive outcomes, which is what multi-class intrusion detection requires when traffic must be classified, such as benign, denial-of-service, probe, and privilege escalation. Its probability mass function must be normalized, meaning every value falls between 0 and 1 and the values across all classes add to exactly 1.

Gaussian (Normal) Distribution. The Gaussian distribution describes continuous quantities and is defined by a mean and a variance (Goodfellow et al., 2016). Measurements such as bytes transferred per session or requests per minute from a given host tend to cluster near a typical value and appear less often further away. This shape makes statistical thresholding possible: under a Gaussian baseline, an observation several standard deviations from the mean is unlikely to occur by chance and is therefore worth flagging.

Model Analysis: How ML Models Use These Concepts

Naive Bayes Classifiers

Naive Bayes applies Bayes' theorem while assuming that features are independent of one another given the class label, so the probability of a class is proportional to the class prior multiplied by the probability of each feature (Zhang et al., 2023, section 22.9). When features are binary word-presence indicators, each of those per-feature probabilities is a Bernoulli distribution estimated by counting how often the word appears in each class.

A phishing filter is a good example showing that structure. Five thousand labeled emails, encoded against a 3,000-word vocabulary, form a $5{,}000 \times 3{,}000$ binary matrix, where each row is an email, and each column indicates whether a given word is present. If the word “verify” appears

in 40% of phishing messages but only 2% of legitimate messages, its presence strongly shifts the result toward the phishing class. Laplace smoothing adds a small constant to every count so that a word never observed in one class does not force the entire calculation to zero.

Logistic and Softmax Regression

Logistic regression computes a linear score $z = Xw + b$ and passes it through the sigmoid function, which maps any real number to the open interval (0, 1). That output is the parameter p of a Bernoulli distribution over the malicious and benign labels. Softmax regression extends the same idea to more than two classes (Zhang et al., 2023). For connection records with 41 features, sorted into 5 attack classes, the weight matrix W has shape 41×5, and the softmax function scales the resulting scores so that they sum to 1, producing a categorical distribution over the 5 classes.

Regularization uses the norms described earlier. Adding the squared L2 norm of the weight vector to the training objective discourages any single feature from dominating the decision, which is useful when a high-variance field, such as packet count, would otherwise overwhelm the remaining features.

Neural Networks

A neural network stacks layers, each of which performs a matrix multiplication followed by an element-wise nonlinearity, computing $H = \sigma(XW + b)$. A malware classifier that takes 500 static features into hidden layers of 128 and 64 units uses weight matrices with shapes 500×128, 128×64, and 64×1. Feeding a batch of 256 samples means passing a 256×500 matrix through three matrix multiplications rather than processing samples one at a time, which is why training on a GPU is practical.

Probability appears at both ends of the network. Weights are initialized by drawing small random values from a Gaussian distribution, which prevents every unit in a layer from learning the same thing. At the output, a sigmoid produces a Bernoulli probability for binary malware detection, and a softmax produces a categorical distribution when the task is to identify which malware family a sample belongs to.

Gaussian Anomaly Detection

Anomaly detection reverses the usual approach by modeling only normal behavior. This matters in practice because labeled attack data is scarce while benign traffic is plentiful. The model estimates a mean and a standard deviation for each feature from clean traffic, which together describe a Gaussian baseline for what normal looks like.

A new observation is then scored by how far it sits from that baseline. Each feature's difference from its mean is first divided by that feature's standard deviation, which puts quantities measured on different scales, such as port numbers and byte counts, onto common footing, and the L2 norm of the resulting vector gives a single distance. An observation several standard deviations from the fitted distribution is improbable and is flagged for review. This model draws on both areas of mathematics: the Gaussian distribution defines what “normal” means, and the norm

measures the distance from it. When the feature set is large, eigen-decomposition via principal component analysis can reduce dimensionality before computing the distance.

Conclusion

Linear algebra supplies both the containers and the arithmetic of applied machine learning. Vectors hold single observations, matrices hold datasets and learned weights, tensors hold batched and sequential data, and matrix multiplication performs thousands of scoring operations in one step. Probability and statistical distributions supply the interpretation, allowing a model to report how confident it is rather than returning a bare verdict, which lets a security team rank alerts instead of treating them all the same. The four models examined here combine the same small set of concepts in different ways, which is why understanding vectors, norms, and the Bernoulli and Gaussian distributions is useful for the rest of the course.

References

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press. https://www.deeplearningbook.org

Wang, Y. (2026). *1.4.1 Linear algebra* [Course slides]. CYSE 420, Old Dominion University.

Zhang, A., Lipton, Z. C., Li, M., & Smola, A. J. (2023). *Dive into deep learning*. Cambridge University Press. https://d2l.ai