The Machine Learning Engineer’s Guide

I. Comprehensive Curriculum Overview

The roadmap for a Machine Learning Engineer requires a rigorous, multi-faceted educational approach that spans theoretical mathematics, pragmatic programming proficiency, foundational data science libraries, and advanced specialization in deep learning and AI disciplines.[1] This curriculum guide outlines a structured, four-phase learning plan, specifically integrating the requested resources: the academic authority of Gil Strang for mathematical foundations, the hands-on coding efficiency of BroCode for rapid Python skills acquisition, and the industry-focused depth of O’Reilly Media for production-ready technical knowledge.

The progression is designed sequentially, ensuring that the necessary prerequisite theory is mastered before implementation, allowing the learner to fully comprehend not just *how* to use algorithms, but *why* they function as they do and *how* to troubleshoot them effectively in production environments.

II. Phase Zero: The Foundational Pillars (Mathematics and Logic)

Phase Zero is mandatory for any serious ML practitioner, as it establishes the quantitative frameworks that govern data representation, optimization, and statistical inference. Without these mathematical pillars, algorithmic work is reduced to mere blind application of external libraries.[1]

1.1. Linear Algebra: The Language of High-Dimensional Data

Linear algebra is the bedrock of machine learning, providing the mathematical apparatus for representing data and transformations.

Primary Resource: Gilbert Strang: *Introduction to Linear Algebra* [2]

Professor Gil Strang's text is a standard reference lauded for its clear, applied approach to subjects like Matrix & Matrix Operations, Scalars, Vectors, Tensors, Determinants, and the Inverse of a Matrix.[1] This resource is critical for developing the proper geometric intuition for data in high-dimensional space. The text's focus on the interconnections between geometry, algebraic systems, and practical applications makes it suitable for engineering students.[3]

A critical area of study within linear algebra is the examination of **Eigenvalues**, **Diagonalization**, and the **Singular Value Decomposition (SVD)**.[1] Understanding SVD is paramount because it provides the mathematical core for Principal Component Analysis (PCA), which is listed in the roadmap under Dimensionality Reduction.[1] PCA, a foundational unsupervised learning technique, relies on SVD to decompose a data matrix into orthogonal components that capture the maximum variance. A thorough comprehension of SVD from a resource like Strang's allows the practitioner to appreciate dimensionality reduction not as a black-box function call, but as a systematic, robust process of data compression and noise filtration based on vector projection and variance optimization. This foundational understanding is necessary when debugging complex models in Phase Two.

1.2. Calculus and Optimization: The Engine of Learning

Calculus provides the tools necessary to define, navigate, and minimize the error functions (Loss Functions) that drive all supervised learning algorithms and neural networks.

Primary Resource (Applied): O'Reilly: *Essential Math for Data Science* (Thomas Nield) [4]

Supplementary Resource (Theory): O'Reilly: Quick Calculus: A Self-Teaching Guide 5
The most essential calculus topics for ML are Derivatives, Partial Derivatives, the Chain Rule, and the Gradient, Jacobian, and Hessian matrices.[1] The

Essential Math for Data Science resource is selected for its focus on integrating these concepts with their immediate applications in data science, linear algebra, and probability theory.[4]

The Chain Rule of differentiation holds singular importance as it is the mathematical mechanism underlying the Backpropagation algorithm, which adjusts the weights in a neural network.[1] Backpropagation computes the gradient—the partial derivatives of the loss function with respect to every weight in the network. Repeated application of the Chain Rule across layers is required to determine how a small change in a weight deep within the network impacts the final output error. Without a strong grasp of the Chain Rule, the fundamental learning mechanism in Deep Learning (Phase III) remains opaque. Furthermore, understanding the Gradient allows the engineer to interpret how optimization algorithms like Gradient Descent navigate the complex loss landscapes of ML models. The Hessian matrix, composed of second partial derivatives, is also required knowledge, particularly for advanced optimization techniques (e.g., Newton's methods) and determining the local convexity of loss functions.[1]

1.3. Probability and Statistics for Inference

Statistical reasoning dictates how data is described, how uncertainty is quantified, and how model performance can be reliably generalized to unseen data.

Primary Resource: O'Reilly/Pearson: *Probability and Statistics for Machine Learning* (Jon Krohn) [6]

This specialized O'Reilly text is ideal because it covers essential topics—Basics of Probability, Bayes Theorem, Random Variables, Probability Density Functions (PDFs), and Types of Distribution [1]—with a direct orientation toward ML applications. The curriculum must transition from descriptive statistics (Graphs & Charts, summarizing data distributions) to inferential statistics.[1]

Inferential statistics forms the foundation for effective **Model Evaluation** in later phases.[1] The objective is not merely calculating metrics like F1-Score or ROC-AUC, but ensuring those results are statistically robust and representative of the model's true performance on the broader population, rather than artifacts of the sample test set. Statistical testing enables the practitioner to quantify confidence in model results and compare competing models rigorously, mitigating the risk of deploying a brittle model. Jon Krohn's resource explicitly addresses these principles within the context of model development.[6]

1.4. Discrete Mathematics for Computational Thinking

Primary Resource: O'Reilly: *Essentials of Discrete Mathematics* (David J. Hunter) [8]

While Calculus and Linear Algebra address the continuous optimization aspects of ML, Discrete Mathematics provides the formal structure for computational logic, data structures, and algorithmic complexity.[1] The O'Reilly guide covers fundamental concepts such as Sets, Relations, Functions, Counting, and Boolean Algebra.[9]

This domain is crucial because it governs the practical efficiency of ML implementation. An ML engineer must understand algorithm performance—how quickly an algorithm executes and how its runtime scales with increasing data size. This performance analysis is rooted in the principles of discrete mathematics.[10] Proficiency here ensures the learner develops strong computational thinking, allowing for the selection of algorithms and data structures that are not only theoretically sound but also scalable for production environments (a necessity for MLOps).

III. Phase One: Programming Mastery and Data Wrangling

Transitioning from abstract theory, this phase focuses on acquiring immediate, practical fluency in Python and its data handling ecosystem, establishing the necessary tools for all subsequent ML work.

2.1. Python Fundamentals and OOP (The BroCode Path)

As requested, the Python programming foundation relies on the effective, hands-on approach provided by online resources.

Primary Resource (Video Course): BroCode: *Python Full Course for free (2024)* [11]

BroCode offers a rapid, comprehensive introduction to Python fundamentals, including Basic Concepts, Variables and Data Types, Loops, Conditionals, Functions, Exceptions, and crucial Object-Oriented Programming (OOP).[1] The advantage of this resource is its ability to deliver immediate programming skill acquisition [User Query].

Mastery of OOP principles is not merely an academic exercise; it is vital for constructing professional, reusable, and maintainable ML codebases. An ML engineer frequently develops custom data transformers, pipelines, or algorithm classes (e.g., custom estimators in Scikit-learn). These components must adhere to software engineering standards, which rely on OOP for encapsulation, inheritance, and modular design. This structured approach, learned early, is essential for transitioning prototypes into scalable, production-ready systems.

2.2. The Scientific Python Stack: NumPy, Pandas, and Data Preparation

The second pillar of implementation requires mastery of the libraries that handle large-scale, array-based data manipulation.

Primary Resource: O'Reilly: *Python Data Science Handbook* (Jake VanderPlas) [13]

Supplementary Resource: O'Reilly: Python Data Analytics: With Pandas, NumPy, and Matplotlib (Fabio Nelli) 14
The *Python Data Science Handbook* serves as the essential reference, comprehensively detailing NumPy (for efficient storage and manipulation of dense data arrays) and Pandas (featuring the DataFrame for labeled/columnar data).[13]

The roadmap places significant emphasis on Data Structures, Data Collection, Data Formats (CSV, Excel, JSON, Parquet), Data Cleaning, and Preprocessing Techniques.[1] In industry, data preparation is typically the most time-consuming task. Pandas is the established tool for operations such as data indexing, handling Missing Data, combining datasets (Merge and Join), and Aggregation and Grouping.[13] A practitioner must be adept at these manipulation techniques because the quality of the input data dictates the ceiling of model performance. Effective data cleaning and wrangling directly lead to successful Feature Engineering in Phase Two.

2.3. Data Visualization, SQL, and Data Sourcing

2.3.1. Data Visualization

Visualization tools are necessary for Exploratory Data Analysis (EDA), which precedes model building.

Resource: Integrated within O'Reilly: *Python Data Science Handbook*.[13]

The Handbook covers Matplotlib and Seaborn, providing capabilities for generating flexible data visualizations.[13] Visualization enables the engineer to perform Descriptive Statistics, understand data distributions, identify outliers, and discover initial correlations, ensuring data quality and informing subsequent feature engineering decisions.[1]

2.3.2. Data Storage and Access (SQL and NoSQL)

An ML Engineer's responsibilities extend beyond model training to securing and accessing real-world data, which often resides in diverse enterprise systems.

Primary Resource (SQL): O'Reilly: *Learning SQL, 3rd Edition* (Alan Beaulieu) [17]

Primary Resource (NoSQL): O'Reilly: Resources covering Amazon DynamoDB or Redis Stack 19
The curriculum explicitly requires knowledge of both SQL and No-SQL databases, as well as accessing data via APIs.[1]

Learning SQL teaches foundational querying, data manipulation techniques, and creating efficient tables.[17] This is vital for querying traditional relational databases (e.g., MySQL, PostgreSQL, SQL Server).

However, modern scalable applications often rely on NoSQL databases for high-performance and flexibility.[19] The engineer must understand how to interact with these different persistence layers, especially when dealing with high-throughput or complex data formats.[1] The requirement for API integration [1] further solidifies the need for database familiarity, as many internal data services expose their data through REST APIs, which often connect directly to these underlying SQL or NoSQL stores.[22]

IV. Phase Two: Core Machine Learning and Production Readiness

This phase utilizes the mathematical and programming foundations to explore classical ML algorithms, focusing heavily on model construction, interpretation, and rigorous evaluation using industry-standard libraries.

The subsequent three phases rely on a singular, definitive text recognized for its blend of theory and practicality: **O'Reilly: *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow (HOML)* (Aurélien Géron)**.[23]

3.1. End-to-End ML Workflow and Feature Engineering

The HOML book guides the learner through the entire project lifecycle, starting with data preparation, Train-Test Data Split, and the crucial step of Feature Engineering.[1]

Key Topics: Feature Scaling & Normalization, Feature Selection, Dimensionality Reduction.[1]

The importance of Feature Scaling (e.g., standardizing or normalizing features) cannot be overstated. Scaling is mathematically required for the efficient convergence of iterative optimization algorithms like Gradient Descent, which is used in Logistic Regression and all neural networks. If features are not scaled, the loss surface will be highly anisotropic, causing the optimization process to oscillate and converge slowly, if at all. This practical necessity connects directly back to the calculus concepts from Phase Zero. The application of dimensionality reduction techniques, particularly PCA, leverages the SVD understanding established earlier, completing the connection between Phase Zero's theory and Phase Two's implementation.[1]

3.2. Supervised Learning: Classification and Regression

This section covers the most widely used algorithms for predictive modeling.

Resource: HOML (Scikit-Learn sections on Classification and Regression) [23]

The curriculum must cover a full spectrum of supervised methods [1]:

- **Regression:** Linear Regression, Polynomial Regression.
- **Classification:** Logistic Regression, Support Vector Machines (SVM), Decision Trees.
- **Ensemble Methods:** Random Forest, Gradient Boosting Machines.

A key topic is **Regularization** (Lasso, Ridge, ElasticNet).[1] Regularization is the process of adding a penalty term to the loss function to prevent overfitting, a major challenge when dealing with complex data. Ridge Regression utilizes the

$\ell 2$ norm to shrink model weights (tending to keep all features), while Lasso uses the $\ell 1$ norm, which can drive some weights to exactly zero, effectively performing automatic feature selection. The concept of vector norms (magnitude of vectors) is a direct application of linear algebra principles, demonstrating how mathematical foundations translate into practical modeling stability and complexity control.

3.3. Unsupervised Learning and Clustering

Unsupervised learning addresses tasks where the training data lacks labels.

Resource: HOML (Unsupervised sections)

Topics include Clustering (K-Nearest Neighbors, Hierarchical, Probabilistic) and Dimensionality Reduction (PCA).[1] A critical component here is the inclusion of

Autoencoders under Dimensionality Reduction.[1] Autoencoders are neural networks designed to learn efficient, compressed representations of data. Their appearance in the Unsupervised Learning section serves as a perfect stepping stone into the complexity of Deep Learning architectures, bridging traditional ML to the methods explored in Phase Three.

3.4. Model Evaluation and Validation Rigor

The ability to build a model is meaningless without the ability to evaluate it rigorously.

Resource: HOML (Model Evaluation sections) [26]

The curriculum mandates mastery of multiple metrics: Accuracy, Precision, Recall, F1-Score, ROC-AUC, and Log Loss, alongside the Confusion Matrix.[1] Relying solely on Accuracy is often deceptive, particularly when dealing with imbalanced datasets (e.g., fraud detection where true positive cases are rare). The engineer must use the Confusion Matrix to understand trade-offs: maximizing Recall is critical when false negatives are costly (e.g., missing a disease), while maximizing Precision is essential when false positives are expensive (e.g., mistakenly rejecting a qualified loan applicant).

Furthermore, rigorous validation techniques such as **K-Fold Cross Validation** [1] are essential for ensuring that model performance metrics are not sensitive to the specific train/test split. Proper validation ensures the model's measured generalization ability is trustworthy before deployment.

Table 1: The ML Engineer Foundational Curriculum: Phase Zero and Phase One

Curriculum Module	Topics Covered (Roadmap)	Primary Resource	Author/Publisher	Focus
Linear Algebra	Vectors, Matrices, SVD, Eigenvalues	Introduction to Linear Algebra	Gil Strang	Academic Theory & Applied Math [2]
Calculus & Optimization	Derivatives, Gradient, Chain Rule	Essential Math for Data Science	Thomas Nield (O'Reilly)	Applied Math for ML [4]
Probability & Statistics	Bayes Theorem, Distributions, Inference	Probability and Statistics for ML	Jon Krohn (O'Reilly)	Statistical Modeling [6]
Python Programming	Basics, Functions, OOP, Exceptions	Python Full Course for free	BroCode	Rapid Hands-On Coding [11]
Data Stack	NumPy, Pandas, Matplotlib, Data Cleaning	Python Data Science Handbook	Jake VanderPlas (O'Reilly)	Data Wrangling & EDA [13]
Databases & APIs	SQL Queries, NoSQL Concepts, Data Sources	Learning SQL, 3rd Edition	Alan Beaulieu (O'Reilly)	Production Data Access [17]

V. Phase Three: Deep Learning and Neural Network Architectures

Deep Learning expands the capabilities of ML using interconnected layers of perceptrons, enabling complex feature extraction for structured and unstructured data (image, text, video).[1] This phase leverages the implementation spine established in Phase Two.

4.1. Neural Network Basics and Training Dynamics

Primary Resource: HOML (Deep Learning chapters) [23] and

O'Reilly: *Neural Networks and Deep Learning* [27]

This section covers the foundational units: the Perceptron and Multi-layer Perceptrons (MLP). Training requires a deep understanding of Forward propagation, Back Propagation, Activation Functions, and Loss Functions.[1] The HOML resource is highly practical, guiding the learner through the use of high-level Deep Learning Libraries such as TensorFlow and Keras.[23]

A critical consideration for the modern ML engineer is framework flexibility. While HOML focuses heavily on TensorFlow, many contemporary industrial and research applications favor PyTorch due to its dynamic computational graph. To ensure versatility, the curriculum must integrate the specialized text: **O'Reilly: *Machine Learning with PyTorch and Scikit-Learn*** (Sebastian Raschka et al.).[24] This supplementary resource ensures proficiency in both major deep learning ecosystems, a necessity for tackling diverse production needs. Advanced training topics are also crucial, including methods for handling Vanishing/Exploding Gradients (like Xavier and He Initialization, and Batch Normalization), and techniques for speeding up training via Faster Optimizers (Momentum, Adam, RMSProp).[1]

4.2. Specialized Architectures (CNNs, RNNs, and Transformers)

Deep learning’s power lies in its specialized architectures designed for specific data types.

Resource: HOML and **O'Reilly: *Neural Networks and Deep Learning*** [27]

1. **Convolutional Neural Networks (CNNs):** CNNs are specialized for spatial data like images and video.[1] The curriculum requires understanding core operations: Convolution, Padding, Pooling, and Strides.[1] These concepts are applied in Image Classification and Image Segmentation, crucial applications detailed in the roadmap.[1]
2. **Recurrent Neural Networks (RNNs):** RNNs, particularly Gated Recurrent Units (GRU) and Long Short-Term Memory (LSTM) cells, are designed for sequential data processing.[1] LSTMs mitigate the vanishing gradient problem in traditional RNNs, making them powerful for time-series analysis and initial sequence-based NLP tasks.
3. **The Transformer Model:** The curriculum must recognize the profound shift toward the **Attention Mechanism** and **Transformers**.[1] Transformers, utilizing Self-Attention and Multi-head Attention [1], have superseded RNNs as the dominant architecture for sequence modeling (including NLP, covered in Phase 5.1). The selected O'Reilly texts cover these advanced architectures, ensuring the learner is focused on state-of-the-art methods essential for modern AI development.[29]

VI. Phase Four: Advanced Domains and Specialization

This concluding phase focuses on high-level specialization areas and the principles necessary for reliable, production-grade AI systems, completing the full ML Engineer profile.[1]

5.1. Natural Language Processing (NLP)

NLP allows machines to process and understand human language, leveraging the sequence modeling skills acquired in Phase 4.2.

Primary Resource: O'Reilly: *Real-World Natural Language Processing* (Masato Hagiwara) [30]

This text emphasizes practical application and deployment.[30] The curriculum must detail the standard NLP Pipeline:

- **Preprocessing:** Tokenization, Lemmatization, and Stemming.[1]
- **Representation:** Understanding **Embeddings** [1], which transform words into dense, context-aware vectors (e.g., Word2Vec, BERT). This concept directly links to the linear

algebra concept of vector representation from Phase Zero.
- **Modeling:** Application of advanced techniques like Attention Models.[1]

The *Real-World NLP* resource guides the practitioner through using modern tools like HuggingFace Transformers, enabling the construction of sophisticated systems like named entity taggers, machine translation systems, and chatbots.[30]

5.2. Reinforcement Learning (RL)

RL is a distinct ML paradigm where an agent learns optimal behavior by interacting with an environment, maximizing a cumulative reward.

Primary Resource: O'Reilly: *Reinforcement Learning* (Phil Winder and Alexander Zai) [31]

The focus of this resource on industrial applications and core RL fundamentals makes it appropriate for the expert-level curriculum.[31] RL begins with the mathematical formalism of

Markov Decision Processes (MDPs).[33] Key algorithms covered include:

- **Value-Based Methods:** Q-Learning and Deep-Q Networks (DQNs).[1] DQNs apply neural networks (Phase III) to estimate Q-values, enabling RL for complex state spaces.
- **Policy-Based Methods:** Policy Gradient methods and sophisticated **Actor-Critic Methods**.[1] Policy gradient methods directly optimize the policy (the agent's behavior strategy), often achieving better convergence properties in continuous action spaces.

The selected text covers cutting-edge deep RL algorithms (PPO, SAC, TD3) and uses industry-standard tools like PyTorch and OpenAI Gym for practical implementation.[31]

5.3. Explainable AI (XAI) and Trustworthy Systems

As ML models become integral to high-stakes decisions, the ability to explain their predictions becomes a professional and regulatory necessity.

Primary Resource: O'Reilly: *Explainable AI for Practitioners* (Michael Munn, David Pitman) [34]

XAI is the practice of designing trustworthy and interpretable ML solutions. This resource shifts the focus from optimizing accuracy to enhancing interpretability and model governance.[35] The text distinguishes between types of explanations (Local vs. Global) and methodologies.[36] Key techniques covered include:

- **Feature Attributions:** Methods like LIME (Local Interpretable Model-agnostic Explanations) and Grad-CAM (Gradient-weighted Class Activation Mapping).[36]
- **Deep Learning Explainability:** XAI techniques specifically applied to complex architectures like DNNs, Keras, and TensorFlow models.[35]

Understanding XAI is crucial for the modern ML engineer working in sensitive domains, ensuring models are not opaque black boxes but are auditable and fair.

5.4. Recommendation Systems

Recommendation Systems (RecSys) are specialized applications crucial for maximizing user engagement across e-commerce and media platforms.[1]

Primary Resource: O'Reilly: *Practical Recommender Systems* [37]

The text covers the core algorithmic approaches: Collaborative Filtering (based on user similarity) and Content-based Filtering (based on item features).[37] Content-based filtering, for example, often utilizes NLP techniques (Phase 5.1), such as TF-IDF, to analyze item descriptions and build user profiles.[37]

Modern recommendation systems are complex production entities. Therefore, the resource also addresses the practicalities of building and deploying these systems, covering relevant MLOps components like PySpark, SparkSQL, and FastAPI.[38] This final specialization topic reinforces the need for integrating data science skills (Phases I and II) with sophisticated deployment strategies required for large-scale production (Phase IV).

Table 2: Core and Advanced ML Reading List (Phase Two, Three, and Four Spines)

ML Domain	Core Algorithms/Concepts	Primary O'Reilly Resource	Implementation Framework	Roadmap Integration
Core ML	Supervised/Un	Hands-On ML	Scikit-learn,	End-to-End

Spine	supervised, Model Evaluation, Ensemble Methods, NN Basics	with Scikit-Learn, Keras, and TensorFlow (HOML)	Keras/TF	ML Workflow [23]
Deep Learning (Alt.)	NNs, CNNs, RNNs, Training Dynamics	Machine Learning with PyTorch and Scikit-Learn	PyTorch	Framework Flexibility [28]
Reinforcement Learning	Q-Learning, Policy Gradients, DRL Algorithms	Reinforcement Learning	Phil Winder/Alexander Zai	Industrial Applications, Gym [31]
Explainable AI (XAI)	LIME, Grad-CAM, Interpretability, Trust	Explainable AI for Practitioners	Michael Munn, David Pitman	Trustworthy AI, Governance [35]
Natural Language Processing	Tokenization, Embeddings, Transformers	Real-World Natural Language Processing	Masato Hagiwara	Practical NLP Pipelines [30]
Recommendation Systems	Collaborative Filtering, Content-Based, MLOps	Practical Recommender Systems	Francesco Ricci	System Design, User Behavior [37]

VII. Conclusions and Recommendations

The creation of an expert-level Machine Learning Engineer curriculum necessitates a careful balance between theoretical mastery and production-level implementation skills. The proposed four-phase curriculum successfully integrates the mandated resources—Gil Strang, BroCode, and O'Reilly—to cover the exhaustive ML roadmap.[1]

Gil Strang's foundational texts anchor the curriculum in rigorous applied mathematics, ensuring that complex concepts like PCA and regularization are understood at the level of linear algebra and calculus, moving the learner past superficial library usage. BroCode provides the necessary rapid induction into Python, ensuring the learner is immediately functional. The extensive use of O'Reilly resources then provides the required depth in the scientific Python stack (Pandas/NumPy), core ML algorithms (HOML spine), deep learning architectures, and specialized disciplines (RL, NLP, XAI).[4]

A crucial recommendation is the mandatory cross-training in deep learning frameworks, specifically supplementing the TensorFlow/Keras focus of the primary HOML spine with the PyTorch expertise found in dedicated O'Reilly PyTorch titles.[24] This deliberate redundancy ensures the ML engineer possesses the framework flexibility demanded by the rapidly evolving industry landscape. Ultimately, this structured progression, linking mathematical origins to algorithmic applications, prepares the learner not just to use machine learning tools, but to design, optimize, and deploy robust, explainable, and scalable AI solutions.

Works cited

1. machine-learning.pdf
2. List of books by author Gilbert Strang - ThriftBooks, accessed September 29, 2025, https://www.thriftbooks.com/a/gilbert-strang/205276/
3. Linear Algebra and It's Applications by Gilbert Strang, accessed September 29, 2025, https://rksmvv.ac.in/wp-content/uploads/2021/04/Gilbert_Strang_Linear_Algebra_and_Its_Applicatio_230928_225121.pdf
4. Essential Math for Data Science - School of Computing e-Library, accessed September 29, 2025, https://soclibrary.futa.edu.ng/books/Essential%20Math%20for%20Data%20Science%20(Thomas%20Nield)%20(Z-Library).pdf
5. Calculus I [Book] - O'Reilly Media, accessed September 29, 2025, https://www.oreilly.com/library/view/calculus-i/9781465454126/
6. Probability and Statistics for Machine Learning[Video] - O'Reilly Media, accessed September 29, 2025, https://www.oreilly.com/library/view/probability-and-statistics/9780137566273/
7. Probability and Statistics [Book] - O'Reilly Media, accessed September 29, 2025, https://www.oreilly.com/library/view/probability-and-statistics/9789332558229/
8. Discrete Mathematics - O'Reilly Media, accessed September 29, 2025, https://www.oreilly.com/library/view/discrete-mathematics/9781299444607/
9. Discrete Mathematics [Book] - O'Reilly Media, accessed September 29, 2025, https://www.oreilly.com/library/view/discrete-mathematics/9789332503441/
10. Discrete Math You Need to Know - Tim Berglund - YouTube, accessed September 29, 2025, https://www.youtube.com/watch?v=CCr-p1tInwQ

11. Bro Code - YouTube, accessed September 29, 2025, https://www.youtube.com/@BroCodez
12. Python Full Course for free - YouTube, accessed September 29, 2025, https://www.youtube.com/watch?v=XKHEtdqhLK8
13. Python Data Science Handbook, accessed September 29, 2025, https://jakevdp.github.io/PythonDataScienceHandbook/
14. Python Data Analytics: With Pandas, NumPy, and Matplotlib - O'Reilly Media, accessed September 29, 2025, https://www.oreilly.com/library/view/python-data-analytics/9781484239131/
15. Python Data Science Handbook [Book] - O'Reilly Media, accessed September 29, 2025, https://www.oreilly.com/library/view/python-data-science/9781491912126/
16. Python for Data Analysis - NIBM E-Library Portal, accessed September 29, 2025, https://nibmehub.com/opac-service/pdf/read/Python%20for%20Data%20Analysis%20_%20data%20wrangling%20with%20Pandas-%20NumPy-%20and%20IPython.pdf
17. SQL in a Nutshell, 4th Edition - O'Reilly Media, accessed September 29, 2025, https://www.oreilly.com/library/view/sql-in-a/9781492088851/
18. Learning SQL by Alan Beaulieu, accessed September 29, 2025, https://www.r-5.org/files/books/computers/languages/sql/mysql/Alan_Beaulieu-Learning_SQL-EN.pdf
19. Skill: NoSQL Databases - O'Reilly Media, accessed September 29, 2025, https://www.oreilly.com/search/skills/nosql-databases/
20. Skill: SQL - O'Reilly Media, accessed September 29, 2025, https://www.oreilly.com/search/skills/sql/
21. NoSQL Database Comparison - ScyllaDB, accessed September 29, 2025, https://www.scylladb.com/learn/nosql/nosql-database-comparison/
22. What would you love to learn in an intermediate / advanced FastAPI book? - Reddit, accessed September 29, 2025, https://www.reddit.com/r/FastAPI/comments/12ziyqp/what_would_you_love_to_learn_in_an_intermediate/
23. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2nd Edition, accessed September 29, 2025, https://www.oreilly.com/library/view/hands-on-machine-learning/9781492032632/
24. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow - Reddit, accessed September 29, 2025, https://www.reddit.com/r/learnmachinelearning/comments/1ln29t4/handson_machine_learning_with_scikitlearn_keras/
25. Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow, accessed September 29, 2025, https://shashwatwork.github.io/assets/files/ml_ebook.pdf
26. Skill: Scikit-learn - O'Reilly Media, accessed September 29, 2025, https://www.oreilly.com/search/skills/scikit-learn/
27. Neural networks and deep learning [Book] - O'Reilly Media, accessed September 29, 2025, https://www.oreilly.com/library/view/neural-networks-and/9781492037354/
28. Skill: Deep Learning - O'Reilly Media, accessed September 29, 2025,

https://www.oreilly.com/search/skills/deep-learning/
29. Machine Learning Algorithms in Depth - O'Reilly Media, accessed September 29, 2025, https://www.oreilly.com/library/view/machine-learning-algorithms/9781633439214/
30. Skill: Natural Language Processing - O'Reilly Media, accessed September 29, 2025, https://www.oreilly.com/search/skills/natural-language-processing/
31. Skill: Reinforcement Learning - O'Reilly Media, accessed September 29, 2025, https://www.oreilly.com/search/skills/reinforcement-learning/
32. Reinforcement Learning - Phil Winder Ph.D. - Google Books, accessed September 29, 2025, https://books.google.com/books/about/Reinforcement_Learning.html?id=SdcHEAAAQBAJ
33. Reinforcement Learning [Book] - O'Reilly Media, accessed September 29, 2025, https://www.oreilly.com/library/view/reinforcement-learning/9781492072386/
34. Explainable AI for Practitioners - Michael Munn, David Pitman - Google Books, accessed September 29, 2025, https://books.google.com/books/about/Explainable_AI_for_Practitioners.html?id=-9mYEAAAQBAJ
35. Explainable AI for Practitioners [Book] - O'Reilly Media, accessed September 29, 2025, https://www.oreilly.com/library/view/explainable-ai-for/9781098119126/
36. Explainable AI for Practitioners (Early Release, Ch1&2/8) 9781098119133 - dokumen.pub, accessed September 29, 2025, https://dokumen.pub/explainable-ai-for-practitioners-early-release-ch1amp2-8-9781098119133.html
37. Practical Recommender Systems [Book] - O'Reilly Media, accessed September 29, 2025, https://www.oreilly.com/library/view/practical-recommender-systems/9781617292705/
38. Building Recommendation Systems in Python and JAX - O'Reilly Media, accessed September 29, 2025, https://www.oreilly.com/library/view/building-recommendation-systems/9781492097983/