■ AI Master Roadmap: Become the Baap of AI ■

Step 0: Prerequisites

■ **What to Learn (detailed):**
- Computer Science basics: how CPU, GPU, RAM, cache, storage interact
- Linux: basic commands (ls, cd, grep, find, chmod, scp), package managers (apt, yum), bash scripting
- Git & GitHub: clone, commit, push, branch, merge, resolve conflicts, pull requests
- Networking basics: HTTP methods, REST APIs, sockets, latency

■ **Reason:** AI runs on Linux servers, deployed with GitHub workflows, scaled on the cloud.
■ **Influence:** Without this, you can't leave Jupyter notebooks.
■ **Weight:** 5%

Step 1: Python Mastery

■ **What to Learn (detailed):**
- Core syntax: operators, indentation, variables, scoping rules
- Data Types: strings (slicing, regex), lists (append, slicing, comprehension), tuples (immutability), sets (union, intersection), dicts (key-value ops, dict comprehension)
- Functions: positional/default args, keyword args, *args and **kwargs, recursion, lambda functions
- OOP: defining classes, __init__, class vs instance variables, inheritance, method overriding, abstract base classes, dataclasses
- Advanced: decorators (logging, timing), generators (yield), iterators (__iter__, __next__), context managers (__enter__, __exit__), type hints (mypy)
- File Handling: read/write (CSV, JSON, pickle, HDF5 with h5py), encoding issues
- Error Handling: try-except-finally, raising custom exceptions
- Logging: logging module (INFO, DEBUG, ERROR)
- Environment Management: pip, venv, poetry, requirements.txt

■ **Reason:** Python is your AI weapon.
■ **Influence:** If you struggle with Python, you'll struggle everywhere.
■ **Weight:** 10%

Step 2: DSA (Data Structures & Algorithms)

■ **What to Learn (detailed):**
- Data Structures:
• Arrays (slicing, 2D arrays, NumPy basics)
• Linked Lists (insert, delete, reverse)
• Stacks & Queues (LIFO/FIFO, deque)
• HashMap/Dictionary (hashing, collisions)
• Trees (binary tree, BST, traversal: inorder, preorder, postorder)
• Heaps (min/max, priority queue with heapq)
• Graphs (adjacency list/matrix, BFS, DFS)
• Tries (prefix tree for NLP autocomplete)
- Algorithms:
• Sorting: quicksort, mergesort, heapsort
• Searching: binary search, hash lookups
• Graph: Dijkstra, Bellman-Ford, A*, Floyd-Warshall
• DP: knapsack, LIS, coin change

• Greedy: Huffman coding, activity selection
• String matching: KMP, Rabin-Karp
- Complexity: analyze time/space, Big-O, amortized complexity

■ **Reason:** AI data is huge — efficiency matters in preprocessing, feature extraction, search.
■ **Influence:** Avoids slow, memory-heavy code in production AI.
■ **Weight:** 8%

Step 3: Math for AI

■ **What to Learn (detailed):**
- Linear Algebra: vectors, dot/cross product, matrix multiplication, transpose, inverse, determinant, rank, eigenvalues/eigenvectors, SVD
- Calculus: limits, differentiation, chain rule, partial derivatives, Jacobians, Hessians, gradient, divergence, Laplacian
- Probability & Stats: random variables, PMF, PDF, CDF, expectation, variance, covariance, distributions (normal, binomial, Poisson, Gaussian), Bayes theorem, hypothesis testing, p-values, confidence intervals
- Optimization: gradient descent, stochastic GD, convex optimization, Lagrange multipliers
- Information Theory: entropy, cross-entropy, KL divergence

■ **Reason:** Math = intuition. Models are applied math.
■ **Influence:** Separates “black box user” from “AI scientist”.
■ **Weight:** 15%

Step 4: Data Tools & Visualization

■ **What to Learn (detailed):**
- NumPy: arrays, broadcasting, vectorization, slicing, linear algebra (np.linalg), random sampling (np.random)
- Pandas: DataFrames, indexing, selection, filtering, groupby, merging, missing data, datetime ops
- Matplotlib: plots, subplots, legends, styles
- Seaborn: heatmaps, distributions, pairplots
- Plotly/Altair: interactive dashboards (optional)

■ **Reason:** 80% of AI = data cleaning & exploration.
■ **Influence:** If you can’t manipulate data, your models fail.
■ **Weight:** 7%

Step 5: Machine Learning Foundations

■ **What to Learn (detailed):**
- Supervised Learning: linear/logistic regression, decision trees, random forest, SVMs
- Unsupervised Learning: k-means, hierarchical clustering, DBSCAN, PCA, t-SNE
- Model Evaluation: precision, recall, F1, confusion matrix, ROC-AUC, cross-validation
- Feature Engineering: scaling (standardization, normalization), one-hot encoding, embeddings, feature selection
- Bias-Variance tradeoff
- Pipelines: Scikit-learn pipelines

■ **Reason:** ML is the foundation of AI.
■ **Influence:** Builds intuition for DL.
■ **Weight:** 10%

Step 6: Deep Learning

■ **What to Learn (detailed):**
- Neural Networks: perceptron, MLP
- Training: forward pass, backpropagation, autograd, loss functions (MSE, cross-entropy)
- Optimizers: SGD, Adam, RMSProp, LR schedulers
- CNNs: conv layers, pooling, ResNet, EfficientNet
- RNNs: vanilla, LSTM, GRU
- Transformers: self-attention, encoder-decoder, GPT/BERT
- Generative Models: GANs, VAEs, diffusion models
- Training Tricks: batch norm, dropout, early stopping, transfer learning

■ **Reason:** DL = backbone of modern AI.
■ **Influence:** From ML engineer → AI master.
■ **Weight:** 20%

Step 7: Specialized AI Domains

■ **What to Learn (detailed):**
- CV: OpenCV basics, YOLO, SSD, U-Net, Mask R-CNN
- NLP: tokenization, embeddings (Word2Vec, GloVe), transformers, HuggingFace APIs
- Speech: Librosa (spectrograms, MFCC), Whisper (speech-to-text)
- RL: OpenAI Gym, Q-learning, DQN, policy gradients, actor-critic
- Graph AI: NetworkX basics, GNNs, PyTorch Geometric

■ **Reason:** Each domain needs specialized methods.
■ **Influence:** Makes you versatile.
■ **Weight:** 15%

Step 8: AI Engineering & Deployment

■ **What to Learn (detailed):**
- APIs: requests, aiohttp
- FastAPI: build async REST APIs
- Flask: simple APIs
- Streamlit/Gradio: interactive ML apps
- Docker: Dockerfile, build, run, push
- Cloud Basics: AWS Sagemaker, GCP Vertex, Azure ML
- Model Optimization: ONNX, TensorRT, quantization

■ **Reason:** Models are useless unless deployed.
■ **Influence:** Moves you to industry-ready AI engineer.
■ **Weight:** 10%

Step 9: Data Engineering + MLOps

■ **What to Learn (detailed):**
- Databases: SQL (joins, indexing), NoSQL (MongoDB)
- Distributed Data: PySpark, Dask
- Pipelines: Airflow, Prefect
- Experiment Tracking: MLflow, Weights & Biases
- Model Versioning: DVC

- Deployment Scaling: Kubernetes (pods, services)

■ **Reason:** AI projects fail without pipelines.
■ **Influence:** Enterprise AI systems engineer level.
■ **Weight:** 7%

Step 10: Advanced AI + Frontier

■ **What to Learn (detailed):**
- LLMs: prompt engineering, fine-tuning, RAG, LoRA, quantization, PEFT
- Multi-Modal AI: CLIP, BLIP
- Explainable AI: SHAP, LIME, Captum
- AI Safety & Ethics: bias detection, adversarial attacks
- Quantum ML (optional): Qiskit basics

■ **Reason:** This is the future of AI.
■ **Influence:** Makes you researcher/innovator.
■ **Weight:** 8%

■ Final Weighted Breakdown

- Prereqs: 5%
- Python: 10%
- DSA: 8%
- Math: 15%
- Data Tools: 7%
- ML: 10%
- DL: 20%
- Specialized AI: 15%
- Engineering/Deployment: 10%
- MLOps: 7%
- Advanced Frontier: 8%