NULLVECTOR
.monster

PHASE 02 — SIGNAL

DAYS 22–42 · CLASSICAL MACHINE LEARNING · LIVE DEPLOYED APP

You finished Phase 1. That means you built 14 Harvard AI projects from scratch and earned two certifications. Phase 2 is where you build something the whole world can use.

Phase 2 teaches you how AI practitioners work. You will use the same tools professionals use at Google, Meta, and Anthropic: Scikit-Learn, PyTorch, and HuggingFace. Every model is built from scratch first so you understand what is really happening. Then you use the library to build it properly. By Day 42 you will have a live application with a public URL that anyone can use — your first real AI product.

You already know more than you think you do. If variables, functions, and loops feel familiar — even a little — you are ready for Phase 2. The concepts get more interesting from here. The difficulty increases gradually. You have already done the hardest part: starting.

// PHASE 2 AT A GLANCE

Duration	**21 days · 3.5 hours per day minimum**
Milestone	**Day 42 — live deployed ML app with a public URL**
Primary course	**Andrew Ng ML Specialization (audit free): https://www.coursera.org/specializations/machine-learning-introduction**

Hands-on	**fast.ai Practical Deep Learning (free): https://course.fast.ai**
Deployment	**HuggingFace Spaces (free): https://huggingface.co/spaces**
Reference	**Scikit-Learn docs: https://scikit-learn.org/stable/**
PyTorch	**PyTorch tutorials: https://pytorch.org/tutorials/**
Your library	**Machine Learning with PyTorch & Scikit-Learn (Packt — your Humble Bundle)**

WEEK 1 · DAYS 22–28 · ML FUNDAMENTALS · FROM SCRATCH FIRST

DAY 022 · WHAT IS ML

Your first real machine learning model.

■ INDUSTRY (15 MIN)

Search: What is machine learning — any clear 5-minute explainer. You should understand most of it now.

■ STUDY (90 MIN) — Andrew Ng Week 1

Enroll and audit free. Watch Week 1: supervised learning, linear regression.

Every video: pause, type the example, run it, then continue.

Key concept: a model learns by minimizing the difference between its predictions and the truth.

→ https://www.coursera.org/specializations/machine-learning-introduction

■ BUILD (90 MIN) — Linear Regression from Scratch

Part 1: implement linear regression manually — NumPy only, no Scikit-Learn.

Part 2: build the same model using Scikit-Learn. Compare results.

Use a real dataset: California housing prices or Boston housing data.

Print: coefficients, mean squared error, R^2 score.

→ https://scikit-learn.org/stable/getting_started.html

✓ DONE WHEN: Linear regression working both from scratch and with Scikit-Learn. Results match.

DAY 023 · CLASSIFICATION

Your first classifier. Real predictions on real data.

■ INDUSTRY (15 MIN)

Search: How does spam filtering work AI — connects directly to today's build.

■ STUDY (90 MIN)

Andrew Ng Week 2: logistic regression and classification.

Key concept: classification predicts a category, not a number.

Understand: sigmoid function, decision boundary, binary cross-entropy loss.

■ BUILD (90 MIN) — Email Spam Classifier

Build a spam vs. not-spam classifier using the SMS Spam Collection dataset.

Download from: https://archive.ics.uci.edu/dataset/228/sms+spam+collection

Use Scikit-Learn: TfidfVectorizer for text features + LogisticRegression.

Evaluate: accuracy, precision, recall, confusion matrix.

Print the 10 words most predictive of spam.

✓ DONE WHEN: Spam classifier built. Accuracy above 95%. Confusion matrix printed and understood.

DAY 024 · GRADIENT DESCENT

The engine behind every AI model you will ever build.

■ INDUSTRY (15 MIN)

Search: Gradient descent visualized — 3Blue1Brown neural network series episode 2.

■ STUDY (90 MIN)

Andrew Ng Week 2 continued: gradient descent in detail.

Key concept: we move in the direction that reduces loss most steeply.

Learning rate: too high = overshoot, too low = too slow. Tune it.

■ BUILD (90 MIN) — Gradient Descent from Scratch

Implement gradient descent manually for linear regression — NumPy only.

Plot the loss curve: x-axis = iterations, y-axis = loss. It should decrease.

Try 3 different learning rates: 0.001, 0.01, 0.1. Plot all 3 on the same chart.

The chart tells you everything about what the learning rate does.

✓ DONE WHEN: Gradient descent implemented. Loss curves plotted for 3 learning rates. Differences explained in log.

DAY 025 · DECISION TREES + RANDOM FORESTS

Ensemble methods and when to use them.

■ INDUSTRY (15 MIN)

Search: Random forests explained simply — any clear visual explainer.

■ STUDY (90 MIN)

Andrew Ng Week 4: decision trees and tree ensembles.

Key concepts: information gain, Gini impurity, overfitting in trees.

Why Random Forests beat single trees: diversity of independent weak learners.

■ BUILD (90 MIN) — 3-Model Comparison

Use the Titanic survival dataset (download from Kaggle: https://www.kaggle.com/c/titanic/data).

Train 3 models on the same data: Decision Tree, Random Forest, Logistic Regression.

Compare: accuracy, training time, cross-validation score.

Print a table showing all results side by side. Which wins and why?

✓ DONE WHEN: 3-model comparison complete. Results table printed. Winner identified with explanation.

DAY 026 · MODEL EVALUATION

How to know if your model is actually good.

■ INDUSTRY (15 MIN)

Search: Overfitting explained simply — a model that memorizes instead of learns.

■ STUDY (90 MIN)

Andrew Ng Week 3: evaluating and improving models.

Key concepts: train/test split, k-fold cross-validation, bias vs variance.

Overfitting: great on training data, poor on new data. Regularization fixes this.

■ BUILD (90 MIN) — Evaluation Toolkit

Build evaluate.py — a reusable toolkit you will use for the rest of the challenge.

Functions: train_test_split(), cross_validate(), plot_confusion_matrix(),

plot_learning_curve(), print_classification_report()

Test it on your Day 23 spam classifier and Day 25 Titanic model.

✓ DONE WHEN: Evaluation toolkit built and tested on 2 previous models. Learning curves plotted.

DAY 027 · PYTORCH INTRO

The framework that powers modern AI.

■ INDUSTRY (15 MIN)

Search: Why PyTorch vs TensorFlow — what AI researchers actually use in 2024 2025.

■ STUDY (90 MIN)

Work through the official PyTorch Basics tutorial.

Key concepts: tensors, autograd, computational graph, neural network layers.

A tensor is just a multi-dimensional array — like NumPy, but with GPU support and autograd.

→ https://pytorch.org/tutorials/beginner/basics/intro.html

■ BUILD (90 MIN) — First PyTorch Neural Net: XOR Problem

The XOR problem: a neural net must learn a function that no linear model can solve.

Build a 2-layer network: input(2) → hidden(4, ReLU) → output(1, sigmoid).

Train it with binary cross-entropy loss and Adam optimizer.

It should solve XOR in under 1000 epochs. Plot the loss curve.

✓ DONE WHEN: PyTorch neural net trained on XOR. Loss reaches near zero. Loss curve plotted.

DAY 028 · WEEK 1 CAPSTONE

End-to-end ML pipeline. Production thinking starts here.

■ INDUSTRY (15 MIN)

Search: What does a machine learning engineer actually do all day.

■ STUDY (30 MIN) — Review

Look back at Days 22–27. What concepts are still fuzzy?

Spend 30 minutes specifically on your weakest area before building.

■ BUILD (2.5 HRS) — Full ML Pipeline

Choose a real dataset from Kaggle or UCI ML Repository.

Build a complete end-to-end pipeline in one Python file:

1. Load and explore data (print shape, dtypes, missing values)
2. Clean and engineer features (handle nulls, encode categories)
3. Train 3 different models
4. Evaluate all 3 with your Day 26 toolkit
5. Save the best model to disk using joblib
6. Load it back and make a prediction on a new example

This is how professional ML engineers work.

✓ DONE WHEN: Full pipeline complete. Best model saved to disk. Can load and predict on new input.

WEEK 2 · DAYS 29–35 · DEEP ML + FAST.AI · FIRST DEPLOYMENT

DAY 029 · FAST.AI LESSON 1

Your first live deployed AI application.

■ INDUSTRY (15 MIN)

Search: fast.ai Jeremy Howard on why top-down learning works better.

■ STUDY (90 MIN) — fast.ai Lesson 1

Watch Lesson 1 in full. Type every code example as you watch.

Key insight: you build and deploy a real model on Day 1. Top-down learning.

The model will classify images — but the architecture works for almost anything.

→ https://course.fast.ai

■ BUILD (90 MIN) — Image Classifier + First Deployment

Follow the fast.ai Lesson 1 notebook to train an image classifier on a dataset you choose.

Create a HuggingFace account and create a new Space.

Deploy your model using Gradio — fast.ai Lesson 1 walks through this step by step.

By the end of today you will have a live URL that anyone can use to classify images.

→ https://huggingface.co/spaces

✓ DONE WHEN: Image classifier deployed to HuggingFace Spaces. Public URL working. Shared with at least one person.

DAY 030 · FAST.AI LESSON 2

Production deployment. Real engineering thinking.

■ INDUSTRY (15 MIN)

Search: HuggingFace story — how open source AI became the standard.

■ STUDY (90 MIN) — fast.ai Lesson 2

Watch Lesson 2: model improvement, data cleaning, production considerations.

Key insight: garbage data produces garbage models. Data quality beats model complexity.

→ https://course.fast.ai

■ BUILD (90 MIN) — Improve and Redeploy

Apply what you learned in Lesson 2 to your Day 29 classifier.

Try: cleaning bad training examples, adding more data, training longer.

Measure the accuracy improvement. Redeploy the improved version.

Update your HuggingFace Space README to document what you changed and why.

✓ DONE WHEN: Improved classifier redeployed. Accuracy measurably better. Changes documented in README.

DAY 031 · SVMS

A different way of thinking about classification.

■ INDUSTRY (15 MIN)

Search: Support vector machines explained visually — any clear animated explainer.

■ STUDY (90 MIN)

Read: Scikit-Learn SVM documentation and user guide.

Key concepts: maximum margin hyperplane, support vectors, kernel trick.

The kernel trick: SVMs can find non-linear boundaries without explicitly computing them.

■ BUILD (90 MIN)

Train an SVM on 3 different datasets: linearly separable, circular, spiral.

Visualize the decision boundary for each using matplotlib.

Compare SVM accuracy vs. logistic regression on each dataset.

The visualizations will show you exactly when SVMs outperform simpler models.

✓ DONE WHEN: SVM decision boundaries visualized for 3 datasets. Comparison with logistic regression complete.

DAY 032 · UNSUPERVISED LEARNING

Finding patterns without being told what to look for.

■ INDUSTRY (15 MIN)

Search: How Netflix recommends movies — the clustering behind recommendation systems.

■ STUDY (90 MIN)

Andrew Ng Week 8: unsupervised learning, k-means clustering, anomaly detection.

Key concept: unsupervised learning finds structure in data without labels.

PCA: reduces many dimensions to 2 or 3 so you can visualize high-dimensional data.

■ BUILD (90 MIN)

Part 1: K-Means from scratch on the MNIST digit dataset. Visualize cluster centers.

Part 2: PCA to reduce MNIST to 2 dimensions. Plot with colors per digit class.

Part 3: Anomaly detection — find the 10 most unusual images in the dataset.

All 3 visualizations committed to GitHub.

✓ DONE WHEN: K-Means clustering, PCA visualization, and anomaly detection complete. All 3 plots committed.

DAY 033 · FEATURE ENGINEERING

The skill that separates good ML engineers from great ones.

■ INDUSTRY (15 MIN)

Search: Feature engineering for machine learning — why data beats algorithms.

■ STUDY (90 MIN)

Read: Kaggle's feature engineering guide — https://www.kaggle.com/learn/feature-engineering

Key insight: a mediocre model with great features beats a great model with mediocre features.

Types: polynomial features, interaction terms, binning, encoding, scaling, target encoding.

■ BUILD (90 MIN)

Take your Day 25 Titanic model. Without changing the model architecture:

Engineer 5 new features: family size, title from name, cabin deck, fare per person, age group.

Train the same Random Forest on the original features vs the engineered features.

Measure and document the accuracy improvement. It should be significant.

✓ DONE WHEN: 5 new features engineered. Model accuracy improved with documentation of each feature's impact.

DAY 034 · GRADIENT BOOSTING

The algorithm that wins Kaggle competitions.

■ INDUSTRY (15 MIN)

Search: Why XGBoost wins everything — gradient boosting explained.

■ STUDY (90 MIN)

Andrew Ng Week 4 continued: boosted trees.

Key concept: boosting builds models sequentially, each one correcting the previous one's errors.

XGBoost vs Random Forest: boosting beats bagging on tabular data almost every time.

■ BUILD (90 MIN)

Install XGBoost: pip install xgboost

Train XGBoost on your engineered Titanic dataset from Day 33.

Compare: Logistic Regression, Random Forest, XGBoost — same data, same evaluation.

XGBoost should win. Understand why by examining feature importances.

✓ DONE WHEN: XGBoost trained. 3-model comparison complete. Feature importances visualized.

DAY 035 · WEEK 2 CAPSTONE

A complete production ML pipeline.

■ INDUSTRY (15 MIN)

Search: How Airbnb / Spotify / Uber uses machine learning internally.

■ STUDY (30 MIN) — Review

Identify your weakest area from Week 2. Spend 30 minutes filling that gap.

■ BUILD (2.5 HRS) — End-to-End Production Pipeline

Choose a new dataset you find genuinely interesting.

Build a complete pipeline with: data loading, EDA, feature engineering,

model selection (try at least 3), hyperparameter tuning with GridSearchCV,

final evaluation, model export.

Write a 300-word README explaining: what the problem is, what you tried,

what worked, what you'd do next.

This is portfolio-quality work. Treat it that way.

✓ DONE WHEN: Production pipeline complete with README. Committed to GitHub as a standalone project.

WEEK 3 · DAYS 36–42 · DEPLOY + POLISH · YOUR FIRST LIVE AI PRODUCT

DAY 036 · BUILD YOUR APP

From model to product.

■ INDUSTRY (15 MIN)

Search: Gradio vs Streamlit for machine learning apps — which to use when.

■ STUDY (60 MIN) — Gradio or Streamlit

Choose one: Gradio (faster to deploy to HuggingFace) or Streamlit (more flexible).

Work through the quickstart tutorial for whichever you choose.

Think about what your app does: what does the user input? What do they get back?

→ https://www.gradio.app/docs/

→ https://docs.streamlit.io/

■ BUILD (2 HRS) — App Interface

Take your best model from Weeks 1–2 (your strongest pipeline or classifier).

Build a clean web interface around it: clear input, clear output, clear explanation.

The interface should work without any coding knowledge from the user.

Test it by asking someone with no technical background to use it.

✓ DONE WHEN: App interface built and tested by a non-technical user. Running locally without errors.

DAY 037 · DEPLOY TO HUGGINGFACE

Your first permanent live URL.

■ INDUSTRY (15 MIN)

Search: HuggingFace Spaces examples — browse what other people have built and deployed.

■ STUDY (45 MIN) — HuggingFace Spaces Guide

Read the Spaces documentation. Understand: app.py, requirements.txt, README.md.

Your deployment needs 3 files minimum: app.py, requirements.txt, README.md.

The README becomes your Space's landing page — write it for a non-technical audience.

→ https://huggingface.co/docs/hub/spaces

■ BUILD (2 HRS) — Deploy

Create a new Space on HuggingFace Spaces: https://huggingface.co/spaces

Push your app files. Watch the build logs. Fix any errors.

Your app should be live at: yourusername.hf.space/your-space-name

Add the live URL to your GitHub README and your daily log.

✓ DONE WHEN: App deployed to HuggingFace Spaces. Live public URL working. URL in GitHub README.

DAY 038 · DEBUG + POLISH

Make it something you are proud to share.

■ INDUSTRY (15 MIN)

Search: What makes a good ML demo — advice from AI practitioners on showing your work.

■ STUDY (30 MIN)

Look at 5 well-regarded HuggingFace Spaces. What do they do well?

Note: clear titles, helpful descriptions, good example inputs, fast responses.

■ BUILD (2.5 HRS) — 20 Real-World Tests + Fixes

Test your deployed app with 20 real inputs — not the examples you used in training.

Document every failure or unexpected output.

Fix the top 3 most important issues.

Add: example inputs built into the interface, a clear description of what the model does, and a note on its limitations (every honest model has limitations).

✓ DONE WHEN: App tested with 20 real inputs. Top 3 issues fixed. Examples and limitations documented.

DAY 039 · TECHNICAL WRITEUP

Explaining your work is as important as doing it.

■ INDUSTRY (15 MIN)

Search: How to write a machine learning project case study — technical writing advice.

■ STUDY (30 MIN)

Read 2 technical blog posts from ML practitioners on Medium or Towards Data Science.

Note their structure: problem → approach → results → lessons → next steps.

■ BUILD (2.5 HRS) — Project Case Study

Write a 500-word technical case study for your deployed app. Structure:

The Problem: what does this model solve and for whom?

The Approach: what data, what algorithm, what features?

The Results: accuracy, performance metrics, real-world test results.

What I Learned: what surprised you? What would you do differently?

What's Next: what would make this better?

Commit it to your GitHub repo as CASE_STUDY.md

This document is what you show in interviews when they ask about your projects.

✓ DONE WHEN: 500-word case study written and committed to GitHub as CASE_STUDY.md.

DAY 040 · STRETCH DEPLOYMENT

A second live app. Breadth matters.

■ INDUSTRY (15 MIN)

Search: AI startup ideas 2025 — think about problems worth solving.

■ STUDY (30 MIN)

Review the Scikit-Learn algorithm cheat sheet:

https://scikit-learn.org/stable/machine_learning_map.html

Choose a different type of problem from your first app: regression instead of classification,

or text instead of tabular, or image instead of text.

■ BUILD (2.5 HRS) — Second App

Build and deploy a second, different ML application.

Different problem type. Different dataset. Different model.

Same standard: deployed to HuggingFace with a public URL and a README.

Two live applications in your portfolio is significantly more compelling than one.

✓ DONE WHEN: Second app deployed. Two live HuggingFace Spaces with public URLs.

DAY 041 · PORTFOLIO UPDATE

Your public record is your resume.

■ INDUSTRY (15 MIN)

Search: How to build a machine learning portfolio that gets you hired 2024 2025.

■ STUDY (30 MIN)

Look at 3 strong ML engineering GitHub profiles. What do they have in common?

Note: clean READMEs, live demos linked, results quantified, code well-commented.

■ BUILD (2.5 HRS) — GitHub README Overhaul

Update your nullvector-journey GitHub README to include:

A one-paragraph description of who you are and what you're building.

Phase 1 section: Harvard CS50 AI certificate, Azure AI-900, projects list.

Phase 2 section: 2 live app URLs, case study link, skills learned.

Your certification badges (GitHub supports Shields.io badges).

A clear photo or avatar — people engage more with profiles that feel human.

This README is the first thing a recruiter sees. Make it count.

✓ DONE WHEN: GitHub README fully updated with Phase 1 and Phase 2 sections, live URLs, and certifications.

DAY 042 · PHASE 2 COMPLETE

Day 42. Your first live AI product exists.

■ INDUSTRY (15 MIN)

Search: AI engineering job market 2025 — what companies are hiring for.

■ REVIEW (45 MIN)

Open both deployed apps. Test them one final time.

Read your case study. Would you be comfortable presenting this in an interview?

If yes: you are ready for Phase 3. If not: spend today fixing the gap.

■ FINAL LOG + PHASE REVIEW (2 HRS)

Write your Phase 2 Complete section in log.md.

List: both live app URLs, case study link, skills learned, lines of code written.

Document your Phase 3 plan: what are you most excited about? Most nervous about?

Push everything to GitHub. Phase 3 — DEPTH — begins tomorrow.

Phase 3 assumes you are comfortable with PyTorch and have deployed real applications.

You are.

✓ DONE WHEN: Both apps live. Case study committed. GitHub updated. Phase 2 log complete. Phase 3 begins tomorrow.

DAY 42 MILESTONE · LIVE ML APPLICATION DEPLOYED — PUBLIC URL

Deployed ML Classifier on HuggingFace Spaces

A working machine learning application with a public URL that anyone can use right now. Not a notebook. Not a script. A real product. You also have a second deployed application, a technical case study, 42 consecutive GitHub commits, and a GitHub profile that shows real, sustained effort. Phase 3 — DEPTH — begins Day 43. Neural networks, backpropagation, and the NVIDIA NCA-AIIO certification.
