Cloud Contraptions
TRAINING & CONSULTING

To discuss this course and customizations:
Call: +1 434-509-6890 or Email: sales@cloudcontraptions.com

# AI and Modern Machine Learning for Software Developers

## Class Duration

35 hours of live training delivered over 5 days.

## Student Prerequisites

- Working experience with Python (writing functions, using packages, working with data structures)
- Familiarity with basic data concepts such as tables, CSV files, and JSON
- No prior machine learning experience or advanced mathematics required

## Target Audience

Software developers, data analysts, and engineers who want to move beyond calling AI APIs and learn how machine learning actually works: from raw data through trained, evaluated, production-ready models. Equally valuable for technical leads evaluating ML solutions, QA engineers testing ML-powered features, and data professionals transitioning into machine learning engineering roles. This course pairs well with follow-on training in LLM application development and AI-assisted software engineering, and teams ready for production-scale ML should continue to Advanced Machine Learning and Data Engineering.

## Description

This hands-on course walks participants through the complete machine learning lifecycle using the tools that dominate the industry today: pandas and Polars for data preparation, scikit-learn for classical modeling, XGBoost and LightGBM for gradient boosting, and PyTorch for an introduction to deep learning and transfer learning with pretrained networks. Participants work with realistic, messy datasets throughout, cleaning and exploring data, engineering features, training and tuning models, evaluating them honestly, and finally integrating them into real applications with FastAPI and ONNX. All hands-on work is done in Google Colab with agentic AI coding assistance integrated throughout: participants use AI agents to generate, explain,

 

debug, and refactor code the way modern ML teams actually work, while building the judgment to evaluate what the agents produce. The emphasis is relentlessly practical: every concept is applied immediately against scenarios drawn from real engineering work, and participants leave with working code they can adapt to their own projects.

## Learning Outcomes

- Explain the machine learning workflow from raw data to deployed model, and when to train versus use a pretrained model or hosted API.
- Prepare real-world datasets for training: cleaning, encoding, scaling, and splitting with pandas and scikit-learn.
- Apply feature engineering techniques and build leakage-free preprocessing pipelines.
- Build classification and regression models with scikit-learn and gradient boosting libraries.
- Evaluate models with appropriate metrics, cross-validation, and diagnostic tools.
- Identify overfitting and tune hyperparameters systematically, tracking experiments with MLflow.
- Describe how neural networks learn, and build and train deep learning models with PyTorch.
- Apply transfer learning with pretrained networks for image and embedding tasks.
- Integrate trained models into applications through model persistence, ONNX export, and FastAPI-based serving.
- Use agentic AI coding tools effectively across the ML workflow: generating data preparation code, diagnosing training issues, and reviewing model code.

## Training Materials

Comprehensive courseware is distributed online at the start of class. All students receive a downloadable MP4 recording of the training.

## Software Requirements

All hands-on work runs in the free tier of Google Colab, which includes built-in agentic AI coding assistance and free GPU access for the deep learning portions. Students need only a Google account and a modern web browser: no local installation required.

 

To discuss this course and customizations:
Call: +1 434-509-6890 or Email: sales@cloudcontraptions.com

## Training Topics

### The Modern Machine Learning Landscape

- AI, machine learning, and deep learning distinctions
- Supervised, unsupervised, and reinforcement paradigms
- Python ML ecosystem: NumPy, pandas, scikit-learn, PyTorch
- Custom models vs. pretrained models vs. hosted APIs
- Agentic AI coding assistance across the model lifecycle

### Exploring and Understanding Data

- Data loading from files, databases, and APIs
- Exploratory data analysis with pandas and Polars
- Distributions, correlations, and target relationships
- Interactive visualization with Plotly
- Data quality assessment: missing values, outliers, duplicates

### Preparing Training Data

- Cleaning and imputation strategies
- Encoding categorical variables
- Feature scaling and normalization
- Train, validation, and test splitting
- Class imbalance and resampling techniques

### Feature Engineering and Pipelines

- Feature selection and importance measures
- Derived, datetime, and text features
- Dimensionality reduction with PCA
- scikit-learn pipelines and column transformers
- Data leakage causes and prevention

### Core Supervised Learning

- Linear and logistic regression families
- Regularization: ridge, lasso, elastic net
- Decision trees and k-nearest neighbors
- Random forests and support vector machines
- Algorithm selection by data size and type

### Evaluating Models

- Baseline models and sanity checks

 

- Regression metrics: MAE, RMSE, $R^2$
- Classification metrics: precision, recall, F1, ROC-AUC
- Cross-validation strategies
- Error analysis and residual diagnostics

**Tuning and Ensembles**

- Overfitting and underfitting diagnostics
- Grid, random, and Bayesian hyperparameter search
- XGBoost, LightGBM, and CatBoost
- Model explainability with SHAP
- Experiment tracking with MLflow

**Unsupervised Learning**

- k-means, DBSCAN, and hierarchical clustering
- Cluster evaluation and selection
- Anomaly and outlier detection
- UMAP and t-SNE for visualization

**Neural Networks with PyTorch**

- Tensors, autograd, and computational graphs
- Network architecture: layers, activations, loss functions
- Training loops, optimizers, and regularization
- Debugging training: learning curves and failure modes
- GPU acceleration in Colab

**Applied Deep Learning and Pretrained Networks**

- Convolutional networks for image tasks
- Transfer learning with pretrained torchvision models
- Fine-tuning vs. feature extraction
- Embeddings for search and similarity

**Shipping Models: Integration and Serving**

- Model persistence and versioning
- ONNX export and inference from C#, Java, and JavaScript
- Model serving with FastAPI
- Batch vs. real-time inference patterns
- Monitoring, drift, and retraining triggers

 