Artificial Intelligence Data Science & Machine Learning Course

This five-day, hands-on course provides you with practical, working experience across the full artificial intelligence and data science stack. Participants move from core data manipulation in Python through foundational and advanced Machine Learning (ML), into Deep Le arning architectures, and finish by building a Retrieval-Augmented Generation (RAG) application using Large Language Models (LLMs). Each day combines focused instruction with a hands-on lab, so participants leave not just understanding the concepts, but having built working pipelines, models, and applications with industry-standard tools. The course assumes prior coding experience and is designed for those looking to upskill into AI and data science work rather than learn to program for the first time. No prior ML, Deep Learning, or GenAI experience is required. This course is concentrated on the data science, ML, Deep Learning, and GenAI concepts and libraries themselves, with labs oriented toward realistic tasks. This live, instructor-led, hands-on course includes labs as part of the curriculum

This course is available either as a private course for your group, or as a public course for individuals. See details at: https://www.agitraining.com/ai-classes/ai-data-science-machine-learning

AGI holds GSA Contract 47QTCA19D003Y

What You Will Learn

- How to clean, manipulate, and analyze data efficiently using Python libraries such as NumPy and Pandas
- The core ML workflow, including problem framing, feature engineering, model validation, and evaluation metrics
- How foundational ML techniques (regression, classification) differ from advanced techniques (ensembles, unsupervised learning, hyperparameter tuning)
- How neural networks work, including the mechanics of training, and the major architectures: Convolutional Neural Networks (CNNs), sequence models (RNNs/LSTMs), and the Transformer
- Why Transformer architecture underlies modern LLMs, and how attention mechanisms changed deep learning
- How LLMs work in practice, including tokens, embeddings, and context windows
- How to write effective prompts and apply prompt engineering techniques to get reliable model output
- How RAG grounds LLM output in external data, and the tradeoffs involved in production GenAI systems

Skills Learned

- Building reusable, production-oriented data pipelines in Python
- Training, evaluating, and tuning classification and regression models with scikit-learn
- Applying ensemble methods and dimensionality reduction techniques to real datasets
- Interpreting model behavior using feature importance and SHapley Additive exPlanations (SHAP)
- Building and training neural networks using TensorFlow/Keras or PyTorch
- Implementing CNNs for image tasks and sequence models for time-series or text data
- Calling LLM APIs programmatically and engineering prompts for consistent, structured output
- Designing and building a working RAG pipeline: chunking, embeddings, vector database retrieval, and generation
- Evaluating GenAI application output for quality, cost, and reliability in a production context

Day 1: Data Science and Python for ML Workflows

Focus: Core data manipulation and analysis using industry-standard Python libraries, at working-developer pace

NumPy for Numerical Computing

- Arrays, vectorization, and broadcasting: writing efficient, non-loop-based code

Pandas for Data Manipulation

- DataFrames/Series, advanced indexing and filtering
- Cleaning, merging, joining, grouping, pivoting, and reshaping datasets
- Performance considerations for larger datasets

Exploratory Data Analysis (EDA)

- Statistical summarization and distribution analysis
- Visualization with Matplotlib/Seaborn for diagnostic, not just presentation, purposes

Data Pipelines

- Ingesting data from Comma-Separated Values (CSV) files, Structured Query Language (SQL) databases, and Application Programming Interfaces (APIs); structuring reusable, reproducible data pipelines
- Preparing datasets for downstream ML use (train/test splits, encoding, scaling)

Hands-on Lab: *Refactor a messy, real-world dataset into a clean, reusable data pipeline ready for modeling*

Day 2: Foundational ML Techniques

Focus: Foundational ML modeling techniques and the standard ML workflow

ML Problem Framing

- Supervised vs. unsupervised; regression vs. classification; choosing the right approach

Regression and Classification

- Linear/polynomial regression, logistic regression, k-Nearest Neighbors (k-NN), decision trees
- Evaluation metrics: Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and R-squared (R^2) for regression; accuracy, precision, recall, F1 score, and Receiver Operating Characteristic/Area Under the Curve (ROC/AUC) for classification

Feature Engineering

- Encoding, scaling, handling categorical and high-cardinality data
- Feature selection and leakage pitfalls

Model Validation

- Cross-validation, bias-variance tradeoff, avoiding overfitting
- scikit-learn pipelines for reproducible workflows

Hands-on Lab: *Build, evaluate, and iterate on a classification pipeline in scikit-learn*

Day 3: Advanced ML Techniques

Focus: Advanced ML modeling techniques, tuning, and interpretability

Ensemble Methods

- Random forests and gradient boosting frameworks such as Extreme Gradient Boosting (XGBoost) and Light Gradient Boosting Machine (LightGBM); bagging vs. boosting

Unsupervised Learning

- Clustering (k-means, hierarchical); dimensionality reduction using Principal Component Analysis (PCA)

Hyperparameter Optimization

- Grid search, random search, and validation strategy at scale

Model Interpretability

- Feature importance and SHapley Additive exPlanations (SHAP) for explainability

From ML to Neural Networks

- Where classical ML breaks down (unstructured data, high-dimensional inputs)
- Conceptual bridge into neural network architectures

Hands-on Lab: *Build and tune an ensemble model; compare against Day 2 baselines and interpret results*

Day 4: Deep Learning, Neural Networks and Architectures

Focus: Neural networks and deep learning architectures

Neural Network Mechanics

- Layers, weights, activation functions, forward/backpropagation, loss functions, optimizers
- Building and training networks in TensorFlow/Keras or PyTorch

Convolutional Neural Networks (CNNs)

- Convolution/pooling operations; image classification workflows

Sequence Architectures

- Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks for sequential and time-series data
- Where sequence models hit their limits, motivating the shift toward attention mechanisms

The Transformer Architecture

- Self-attention, positional encoding, and why Transformers displaced RNNs for language tasks
- Conceptual on-ramp into how LLMs are built on this architecture

Training in Practice

- Regularization, batch normalization, Graphics Processing Unit (GPU) acceleration, and training-time tradeoffs

Hands-on Lab: *Train and evaluate a neural network on an image or text classification task*

Day 5: GenAI, Large Language Models (LLMs), Prompt Engineering, and Retrieval-Augmented Generation (RAG)

Focus: Practical application of LLMs, prompt engineering, and RAG

How LLMs Work in Practice

- Tokens, embeddings, and context windows: the mental model needed to build with LLMs
- Landscape of model providers and API access patterns

Prompt Engineering

- Prompt structure, system/user roles, few-shot examples
- Chain-of-thought, structured output, and iterative prompt evaluation

Building with LLM APIs

- Programmatic API calls, parameter tuning (temperature, max tokens), handling structured responses

Retrieval-Augmented Generation (RAG)

- Embeddings and vector databases (Chroma/FAISS/Pinecone)
- Designing a RAG pipeline: chunking strategy, retrieval, re-ranking, generation

- Evaluating RAG output quality and reducing hallucination

Production Considerations

- Cost/latency tradeoffs, data privacy, and where RAG fits into existing engineering systems

Hands-on Lab: *Build a working RAG application over a custom document set, from ingestion through query response*

Tools and Technologies Covered

This course can be adapted to your specific stack. In addition to Python, Jupyter, NumPy, and Pandas, it can also include Matplotlib/Seaborn, scikit-learn, XGBoost, TensorFlow/Keras or PyTorch, an LLM API (e.g., OpenAI/Anthropic), and a vector database (e.g., Chroma/FAISS/Pinecone).