Coincent 3-Year Program in Artificial Intelligence with Python
Partnered by Worisgo

Year 1: Live Industrial Training – Build Your Foundation

Gain hands-on industry exposure from day one with 2.5 months of live training in a professional environment. Learn the latest tools and technologies through skill-focused sessions, guided by expert mentors from the industry

Artificial Intelligence Curriculum

Chapter 1: Introduction to Artificial Intelligence

1.1 Python Overview

- Goal: Bring all students to a baseline in Python.
- Covers:
    - What is Python and why it's used in AI
    - Syntax basics: variables, loops, functions
    - Writing a basic script (e.g., calculator or simple data handler)

1.2 Python Intro and Installation

- Step-by-step instructions to:
    - Download Python (Windows, macOS, Linux)
    - Set up IDEs (PyCharm, VS Code, or Jupyter Notebooks)
    - Install packages using pip
    - Check system path and virtual environments

1.3 Basics: Numbers, Strings

- Arithmetic operations (+, -, *, /, %, //)
- String manipulation:
    - Slicing, concatenation, formatting (f-strings)
    - String methods (e.g., .lower(), .split(), .replace())

1.4 Data Types

- Primitive: int, float, str, bool
- Non-primitive: list, tuple, set, dict
- Type casting: int("5"), float(4)
- type() and isinstance() functions

Chapter 2: Introduction to Python Programming

2.1 Python Language Overview

- History of Python and its growing ecosystem
- Comparison with other languages (Java, C++)
- Syntax overview and use cases in data science and AI

2.2 Introduction to Python Programming

- Writing a complete program from scratch
- print(), input(), basic variables and logic
- Emphasis on code structure and readability

2.3 Basics of Python Programming

- If-else conditions
- Nested statements
- While and For loops
- Logic and comparison operators

2.4 Python Programming – Part 1

- Lists: creation, access, loop through
- Tuples: immutability and usage
- Dictionary: keys and values, looping
- List comprehensions and nested structures

2.5 Python Programming – Part 2

- Functions: def, arguments, return values
- Modules: creating and importing (math, random)
- File I/O: reading and writing text files
- Try-except blocks for error handling

Chapter 3: Introduction to Matplotlib (Data Visualization)

3.1 Introduction to Matplotlib

- Purpose: Visualizing trends and comparisons
- Introduction to pyplot, figure, axes

3.2 Matplotlib Plots

- Line, scatter, bar, histogram, pie charts
- Understanding x and y axes, legends, and grid

3.3 Matplotlib Styles – Part 1

- Customizing:
    - Title, x/y axis labels
    - Font size, type, color
    - Line styles (--, -, :)

3.4 Matplotlib Styles – Part 2

- Themes: seaborn styles, ggplot, bmh
- Using plt.style.use()
- Saving plots as images (.png, .svg)

3.5 Matplotlib Plot Variants

- Subplots: multiple plots in one figure
- Plotting from Pandas DataFrames
- Adding annotations, text, and markers

Chapter 4: Introduction to NumPy (Numerical Computing)

4.1 Introduction to NumPy

- Why use NumPy over regular Python lists
- Efficiency and memory handling

4.2 NumPy Data Types and Operations

- Types: int32, float64, bool
- Element-wise operations
- Comparisons and logical operations

4.3 NumPy Arrays

- Creating arrays: np.array, np.zeros, np.arange, np.linspace
- Reshaping: .reshape(), .flatten()
- Indexing and slicing multidimensional arrays

4.4 Array Attributes

- .shape, .ndim, .dtype, .size
- Transpose: .T
- Copy vs View behavior

4.5 NumPy Math

- Basic: sum, mean, std, min, max
- Linear algebra: dot product, matrix multiplication
- Random number generation: np.random

Chapter 5: Introduction to Pandas (Data Handling)

5.1 Introduction to Pandas

- Understanding Series and DataFrames
- Structure of tabular data

5.2 Reading Data

- read_csv(), read_excel(), read_json()
- Handling missing values: NaN, fillna(), dropna()

5.3 Data Manipulation – Part 1

- Selecting rows/columns using loc and iloc
- Adding/removing columns
- Filtering data using conditions

5.4 Data Manipulation – Part 2

- groupby(), aggregation
- Sorting by values or index
- Pivot tables and reshaping

5.5 Data Manipulation – Part 3

- Merging and joining DataFrames
- concat(), merge(), join()
- Time series basics (resample, rolling)

Chapter 6: Machine Learning – Unsupervised Learning

6.1 Introduction to Machine Learning

- What is ML, and its use in real-world applications
- Types of learning: Supervised, Unsupervised, Reinforcement
- Algorithms: Clustering, Classification, Regression

6.2 Unsupervised Learning Techniques

- **Clustering:**
    - K-Means, Hierarchical, DBSCAN
    - Visualizing clusters using PCA or t-SNE
- **Dimensionality Reduction:**
    - PCA, LDA
    - Feature selection and extraction

Chapter 7: Introduction to TensorFlow and Deep Learning

7.0 Getting Started with TensorFlow

- Installing TensorFlow: Install using pip install tensorflow. It supports CPU and GPU versions.
- Tensors: Core data structure in TensorFlow—multi-dimensional arrays used in computations.
- Architecture: TensorFlow uses a computation graph; in v2.x, eager execution allows real-time operation.
- Basic Operations: Includes tensor creation, arithmetic (add, multiply), reshaping, and reduction (sum, mean).

7.1 Introduction to Keras

- Keras Overview: High-level API within TensorFlow for building neural networks easily.
- Building Models: Use Sequential or functional API to define layers; compile with optimizer, loss, and metrics.
- Training & Evaluation: Use fit() to train, evaluate() for test performance, and predict() for outputs.

7.2 Clustering

- Model Building: Use Scikit-learn for classic clustering (K-Means, DBSCAN) or Keras with autoencoders.
- Visualization: Use PCA or t-SNE with matplotlib to plot clustered data in 2D.
- Evaluation: Silhouette Score measures how well data points fit within clusters (closer to 1 is better).

7.3 Deep Learning Basics

- Understanding:
    - Perceptrons, hidden layers, weights, biases
    - Activation functions (ReLU, Sigmoid, Softmax)
- Backpropagation and gradient descent

7.4 Natural Language Processing (NLP)

- Text cleaning (lowercase, punctuation removal)
- Tokenization, stemming, lemmatization
- Word embeddings: Word2Vec, TF-IDF
- Simple NLP tasks: Sentiment Analysis, Spam Detection

7.5 Reinforcement Learning

- Concepts: Agent, Environment, Reward, Policy
- Introduction to Q-Learning and Markov Decision Processes

Year 2: Real-Time Projects – Apply What You've Learned

Transform your knowledge into real-world experience by working on 8 industry-level projects that build your technical and professional skills. Each project enhances your portfolio, strengthening your resume and showcasing your practical abilities. You'll also collaborate in teams, gaining valuable experience in communication, teamwork, and project management—just like in a real work environment.

PROJECTS

News Classification using Natural Language Processing (NLP) is a project that involves automatically categorizing news articles into predefined categories such as sports, politics, technology, or entertainment. It leverages NLP techniques to preprocess textual data—like tokenization, stop word removal, and stemming—followed by machine learning algorithms such as Naive Bayes or Support Vector Machines for classification. The model learns patterns from labeled training data to predict the category of new, unseen news articles. This project is widely used in media monitoring, personalized news feeds, and content organization.

Text Classification with TensorFlow is a process of automatically assigning categories to text data using deep learning models. Leveraging TensorFlow and Keras, you can build neural networks such as CNNs, RNNs, or LSTMs to learn patterns in text and classify it into labels like spam/ham, sentiment (positive/negative), or topic categories. The text is first preprocessed and converted into numerical form using tokenization and embeddings before training the model. This technique is widely used in applications like sentiment analysis, email filtering, and content moderation.

Handwritten Digit Classification is a classic computer vision task where an AI model is trained to recognize digits (0–9) from images, typically using the MNIST dataset. Convolutional Neural Networks (CNNs) are commonly used in this project to automatically learn features like edges, shapes, and curves from the input images. The model is trained on thousands of labeled digit images and can then accurately classify new handwritten digits. This application is widely used in postal automation, bank cheque processing, and form digitization.

Recognition of Objects in AI involves detecting and classifying objects within images or video frames using deep learning techniques. Models like Convolutional Neural Networks (CNNs), YOLO (You Only Look Once), and Faster R-CNN are commonly used for identifying multiple objects with high accuracy and speed. Object recognition is crucial in applications such as autonomous vehicles, surveillance systems, and medical imaging. The process includes training models on large annotated datasets to learn patterns and features specific to different object classes.

Handwritten Digit Classification is a classic computer vision task where an AI model is trained to recognize digits (0–9) from images, typically using the MNIST dataset. Convolutional Neural Networks (CNNs) are commonly used in this project to automatically learn features like edges, shapes, and curves from the input images. The model is trained on thousands of labeled digit images and can then accurately classify new handwritten digits. This application is widely used in postal automation, bank cheque processing, and form digitization.

Automatic Speech Recognition (ASR) is an AI technology that converts spoken language into written text. It uses deep learning models such as Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTM), or Transformer-based architectures to analyze audio signals and transcribe speech accurately. ASR systems are used in voice assistants, transcription services, and real-time communication tools. The process involves feature extraction from audio, acoustic modeling, and decoding using language models.

Landmark Detection in AI involves identifying specific key points or features in images, such as the corners of the eyes, tip of the nose, or joints in the human body. It uses deep learning models like Convolutional Neural Networks (CNNs) to accurately locate these points, even in complex backgrounds or varying poses. This technique is widely used in facial recognition, gesture tracking, augmented reality, and medical imaging. The model is trained on annotated datasets to predict the coordinates of landmarks.

Vision Transformer (ViT) is an advanced deep learning model that applies the Transformer architecture, originally developed for natural language processing, to image recognition tasks. Unlike traditional CNNs, ViT divides images into fixed-size patches, flattens them, and treats them as a sequence of tokens—similar to words in a sentence. Each patch is embedded and passed through Transformer encoders that use self-attention mechanisms to capture global relationships. This allows ViT to learn long-range dependencies across an image more effectively than CNNs. Vision Transformers have shown competitive or superior performance on benchmarks like ImageNet when trained on large datasets. They are particularly powerful in tasks requiring a global understanding of the visual scene. ViT models are scalable and benefit significantly from large-scale pretraining

Year 3 – Placement & Internship Phase:

In the 3rd year of Coincent's program, students are guaranteed an internship with partner companies, complete with a formal Internship Offer Letter and a Completion Certificate upon successful completion. This internship is a complimentary part of the 3-Year "Industrial Training + Internship" model, which also includes live classes, expert mentorship, and hands-on project work. This phase bridges academic learning with real-world application, providing students with valuable professional exposure before graduation.

Coincent also offers structured placement preparation to ensure students are job-ready. This includes portfolio building through 8 real-time projects, certifications aligned with Microsoft standards, and dedicated training for interviews. From mock interviews to resume reviews and HR/technical round prep, every element is designed to transition students from classroom learning to career success. By the 4th year, students are equipped not just with knowledge, but with experience, credentials, and confidence to enter the workforce.