Python and Machine Learning – Course Outline

General points

- Using PAR style delivery – Short video content and suggested exercises
- 45 mins workshop – 15 mins shared group walkthrough exercise 1 – 60 mins 4 additional questions with support.
- Remember that we can use existing videos from ICTLC Python if applicable OR any suggested online video content (Uni Michighan – Python for everybody) or UCC paid courses (please view for content breakdown)

Week 1 & 2: Foundations of Python

Goal: Give business/non-programmer entrants enough Python fundamentals to navigate data analysis and basic ML tasks.

Topics

- Python Basics:
    - Week 1: Variables, data types (including lists, dictionaries and tuples), conditional and arithmetic operators, conditionals
    - Week 2: string manipulations, loops, functions (basic functions like append, sort, etc.), classes, I/O (read/write to files).
- Introduction to IDEs / Notebook Environments:
    - Google Colab setup (notebook cells and structure).
    - VS Code basics (optional, but show how a typical Python project is structured).
- Basic troubleshooting and debugging in Python.

Activities

- Write a few simple scripts (e.g., data cleaning tasks, printing analytics on small text files).
- Practice Jupyter/Colab notebooks with short exercises (e.g., "calculate factorial of a number," "read in a CSV and print first few rows," etc.).

Week 3: Introduction to Machine Learning

Goal: Give a high-level view of ML concepts and make students comfortable with data manipulation libraries.

Conceptual Topics

- ML vs. DL vs. AI (broad overview).
- Types of learning: Supervised, Unsupervised, Reinforcement.
- Basic statistical concepts for data manipulation (mean, median, mode).
- Data augmentation (mention conceptually, especially because it is relevant to future neural network examples).

Practical / Libraries

- **NumPy**: matrix operations, array manipulation, quick linear algebra for ML.
- **Pandas**: data loading, cleaning, and preprocessing (reading CSVs, dataframes, etc.).

Activities

- Load a small dataset from CSV with Pandas, display head/tail, check shape, basic descriptive statistics.
- Practice simple NumPy operations (matrix creation, dot products, slicing).

Week 4: Data Visualisation and Simple ML

Goal: Show students how to visualize data, then perform a basic supervised ML task in scikit-learn.

Topics

- **Data Visualization**:
  - Why we visualize (domain examples).
  - Matplotlib (line plots, scatter plots, histograms).
  - Seaborn (more aesthetically pleasing, advanced plots).
- **Basic ML Workflow with scikit-learn**:
  - Splitting data: training/test sets (and mention validation sets).
  - Simple ML algorithms: e.g., linear regression or a simple classifier (logistic regression or k-NN).
  - Evaluate a model (basic metrics such as accuracy or RMSE).

Activities

- Visualize small dataset relationships (scatter plots, correlation heatmaps).
- Implement a simple regression/classification pipeline in scikit-learn:
  - Load data with Pandas.
  - Split train/test.
  - Train a basic model.
  - Evaluate performance (accuracy or MSE).

Week 5: Neural Networks (Conceptual + A Simple Implementation)

Goal: Introduce neural networks conceptually; do **one** simple end-to-end neural network example to solidify the idea.

Conceptual Topics

- Biological inspiration for NNs (brief).
- Building blocks of a NN:
    - Neurons, layers (input, hidden, output), weights, biases.
    - Activation functions (ReLU, sigmoid) – **focus on 1–2 main ones** rather than too many.
- Loss function & optimizers at a high level.
- Why splitting data is important for NN training (train vs. test vs. possibly validation).

Implementation

1. **High-level**: Perceptron, Use scikit-learn's MLPClassifier or MLPRegressor to build a *very* simple neural network.
2. **To introduce PyTorch**:
    - Show them how to define a simple one-layer or two-layer network class.
    - Forward pass and setting up a training loop.

Activities

- Pick one simple dataset (regression or classification).
- Implement a *single-hidden-layer* neural network.
- Show how to train it and visualize loss over epochs
- **Keep activation function experiments optional or minimal**—perhaps show them how changing from ReLU to sigmoid can impact training quickly, but don't force them to test every possible function.

Week 6: Training and Evaluating Neural Networks

Goal: Reinforce the end-to-end process of training a NN and introduce performance/regularization topics.

Topics

- Overfitting vs. Underfitting (visual examples).
- Regularization: dropout, L2 (conceptual + show code snippet).
- Training process (expanded):
    - Forward pass, backprop, gradient descent (more detail than in Week 5).

    - Loss functions: MSE vs. cross-entropy.
    - Model evaluation & metrics: accuracy, precision, recall, confusion matrix (for classification).

Activities

- Use a slightly deeper network (2–3 hidden layers) for a binary classification (e.g., Iris dataset).
- Train it, evaluate on test set.
- Add dropout (or L2 regularization) and compare performance/overfitting.
- Generate confusion matrix & discuss how to interpret metrics.

Week 7: Diving into Neural Networks

Goal: Provide a high-level overview of advanced network types and let students integrate everything in a mini project.

Topics

- CNNs (Convolutional Neural Networks) – typical use-cases (images).
- RNNs (Recurrent Neural Networks) – typical use-cases (text/time series).
- Show some example architectures or code snippets, but keep it fairly high-level since it is advanced.

Mini-Project (activity)

- "Putting it all together": Students pick a small dataset or we provide a slightly bigger dataset.
- They must do:
    - Data loading + cleaning + visualization.
    - A neural network build (using scikit-learn's MLP or PyTorch).
    - Training, evaluation, basic tuning.
    - Present results.