**Chương 3 – Crash Course on Python, NumPy, Matplotlib, Pandas**

**Mục tiêu:** Giới thiệu các công cụ cơ bản trong Python như NumPy, Matplotlib, và Pandas, giúp người học làm quen với cách thức thao tác với dữ liệu và trực quan hóa dữ liệu.

**1. Python Cơ Bản cho Machine Learning**  
• Tạo biến, danh sách (list), mảng (array).  
• Sử dụng vòng lặp, hàm (function) và cấu trúc điều kiện (if).  
• Định nghĩa và gọi hàm cơ bản trong Python.

**2. NumPy – Thư viện tính toán số học**  
• Tạo mảng từ danh sách với numpy.array().  
• Truy cập dữ liệu bằng chỉ số, hàng, cột, slicing.  
• Thao tác số học giữa các mảng: cộng, trừ, nhân, chia trực tiếp.  
• Hiệu quả cao khi xử lý dữ liệu lớn.

**3. Matplotlib – Thư viện trực quan hóa**  
• Vẽ biểu đồ đường (line plot) bằng plt.plot().  
• Vẽ biểu đồ phân tán (scatter plot) bằng plt.scatter().  
• Tùy chỉnh nhãn trục, tiêu đề và hiển thị với plt.show().

**4. Pandas – Thư viện xử lý dữ liệu**  
• **Series**: Dữ liệu một chiều có nhãn (giống như mảng có tên hàng).  
• **DataFrame**: Dữ liệu hai chiều giống bảng (có tên hàng và tên cột).  
• Truy cập dữ liệu bằng nhãn hoặc cú pháp .ten\_cot.  
• Dễ dàng xem, lọc, thao tác và phân tích dữ liệu.

**Chương 7 – Prepare Your Data For Machine Learning**

**Mục tiêu:** Cung cấp các kỹ thuật tiền xử lý dữ liệu quan trọng như chuẩn hóa, chuẩn hóa phân phối, và nhị phân hóa để làm sạch và chuẩn bị dữ liệu cho mô hình học máy.

1. Tiền xử lý dữ liệu là bước quan trọng vì hầu hết các thuật toán không làm việc tốt với dữ liệu thô.
2. **Rescaling** (chuẩn hóa về khoảng [0, 1]) giúp các thuật toán tối ưu hoạt động chính xác hơn.
3. **Standardization** (chuẩn hóa phân phối) giúp dữ liệu có trung bình bằng 0 và độ lệch chuẩn bằng 1.
4. **Normalization** (chuẩn hóa từng hàng) thích hợp với các thuật toán dựa trên khoảng cách như k-NN.
5. **Binarization** (nhị phân hóa) chuyển đặc trưng thành dạng 0/1, hữu ích cho các thuật toán cần dữ liệu rời rạc.
6. Cần thử nghiệm nhiều kỹ thuật tiền xử lý để chọn phương pháp phù hợp với tập dữ liệu và thuật toán.

**Chương 11 – Spot-Check Các Thuật Toán Phân Loại**

**Mục tiêu:** Hướng dẫn cách thử nghiệm và đánh giá các thuật toán phân loại, giúp người học tìm ra thuật toán phù hợp nhất cho bài toán của mình.

**Các thuật toán phân loại**:

1. **Spot-Checking**: Thử nghiệm với nhiều thuật toán khác nhau để xác định thuật toán phân loại tốt nhất cho bài toán.
2. Các thuật toán phân loại:  
   • **Logistic Regression** (tuyến tính) dùng cho phân loại nhị phân.  
   • **Linear Discriminant Analysis (LDA)** (tuyến tính) cho phân loại nhị phân hoặc đa lớp.  
   • **k-Nearest Neighbors (KNN)** (phi tuyến tính) phân loại dựa trên khoảng cách.  
   • **Naive Bayes** (phi tuyến tính) dựa trên xác suất với giả định các đặc trưng độc lập.  
   • **CART (Classification and Regression Trees)** (phi tuyến tính) sử dụng cây quyết định.  
   • **Support Vector Machines (SVM)** (phi tuyến tính) phân loại bằng cách tìm siêu phẳng phân chia các lớp.
3. **KFold Cross-Validation**: Sử dụng 10 lần gập để đánh giá mô hình, giúp giảm overfitting.

**Chương 15 – Ensemble Learning**

**Mục tiêu:** Trình bày các kỹ thuật học máy nâng cao như Ensemble Learning, giúp cải thiện độ chính xác của mô hình thông qua việc kết hợp nhiều mô hình con.

1. **Tổng quan về các phương pháp Ensemble**:

* **Bagging**: Xây dựng nhiều mô hình từ các mẫu con khác nhau của bộ dữ liệu huấn luyện, sau đó kết hợp các dự đoán để tạo ra kết quả cuối cùng.
* **Boosting**: Xây dựng nhiều mô hình theo chuỗi, mỗi mô hình mới học cách sửa lỗi của mô hình trước đó.
* **Voting**: Kết hợp các dự đoán từ nhiều mô hình (thường là các loại khác nhau) bằng cách sử dụng các thống kê đơn giản như tính trung bình.

1. **Các thuật toán Bagging**:

* **Bagged Decision Trees**: Sử dụng nhiều cây quyết định được huấn luyện trên các mẫu bootstrapped khác nhau của bộ dữ liệu huấn luyện.
* **Random Forest**: Một biến thể của bagging, trong đó, ngoài việc bootstrapping, còn có việc lựa chọn ngẫu nhiên một tập hợp con các đặc trưng tại mỗi điểm phân chia trong các cây quyết định để giảm sự tương quan giữa các cây.
* **Extra Trees**: Một dạng bagging khác, trong đó các điểm phân chia được chọn từ các đặc trưng ngẫu nhiên, dẫn đến nhiều sự ngẫu nhiên hơn và thường cải thiện hiệu suất.

1. **Các thuật toán Boosting**:

* **AdaBoost**: Điều chỉnh trọng số của các mẫu, tập trung nhiều hơn vào các mẫu khó phân loại, và kết hợp các mô hình yếu theo chuỗi.
* **Stochastic Gradient Boosting**: Sử dụng phương pháp giảm dần gradient để tối ưu hóa mô hình bằng cách liên tiếp thêm các mô hình cải thiện các mô hình trước.

1. **Voting Ensemble**:

* Kết hợp các dự đoán từ các mô hình khác nhau (ví dụ: Logistic Regression, Decision Trees, SVM) bằng cách tính trung bình các dự đoán của chúng. Đây là một cách đơn giản và hiệu quả để cải thiện hiệu suất mô hình mà không cần phải tinh chỉnh tham số phức tạp