Practical Work 4: Robust Classification and Uncertainty Modeling

Louis Fippo Fitime*, Claude Tinku, Kerolle Sonfack
Computer Engineering Department, ENSPY

December 24, 2025

TP Objective

- Master the inputs, outputs, and underlying decision boundaries of classification models (Binary and Multiclass).
- Implement and scale core classification algorithms: Logistic Regression, Naive Bayes, KNN, Decision Trees, and Support Vector Machines (SVM).
- Improve model performance using Ensemble Methods (Boosting) and scalable training (Stochastic Gradient Descent/Ascent).
- Critically evaluate models using advanced metrics (Precision-Recall, AUC).
- Explore model robustness and the handling of label uncertainty, directly linking classical ML to the principles of **Evidential Deep Learning (EDL)**.
- Practice Machine Learning Engineering (**MLOps**) for model versioning and tracking.

---

1 Case Study: Robust Medical Image Classification with Uncertainty

This TP aims to develop robust classification variants to address the challenge of predicting disease from medical images, specifically tackling the issue of uncertain or noisy labels, as studied in the paper: *"Robust Medical Image Classification with Uncertain Labels via Evidential Deep Learning"*.

Goal 1. The final objective is to produce a comprehensive report, comparing classical robust models with the conceptual foundation of Evidential Deep Learning (EDL), on medical image datasets known for label uncertainty.

*Corresponding Author: louis.fippo@univ-yaounde1.cm

1.1 Phase I: Data Preparation and Probabilistic Baselines

- **Task 1.1: Data and Preprocessing:**
  1. Load a suitable medical image dataset (e.g., a subset of CheXpert, or a simplified synthetic equivalent for demonstration, using extracted features if necessary to avoid complex DL setup).
  2. Describe the input and output (binary/multiclass nature) of the classification task.
  3. Implement techniques for handling missing data (if any) and perform feature scaling.
- **Task 1.2: Logistic Regression (Large-Scale):**
  1. Implement a **Logistic Regression** model for both binary and multiclass tasks.
  2. Scale the training process using **Stochastic Gradient Ascent (SGA)** to demonstrate training efficiency on large-scale data.
  3. Discuss how the sigmoid function provides a probabilistic interpretation.
- **Task 1.3: Naive Bayes and KNN:**
  1. Implement the **Naive Bayes Classifier**. Justify the independence assumption and its computational advantages.
  2. Implement the **K-Nearest Neighbors (KNN) Classifier**. Discuss the role of K and the challenge of high dimensionality in this model.

1.2 Phase II: Boundary-Based and Non-linear Classification

- **Task 2.1: Decision Trees and Boosting:**
  1. Implement a **Decision Tree** classifier. Analyze the feature importance provided by the model.
  2. Explain the Decision Tree Algorithm (e.g., CART or ID3) and discuss its complexity ($\mathcal{O}(n \cdot m \cdot \text{depth})$).
  3. Improve the tree performance using a **Boosting** method (e.g., AdaBoost or Gradient Boosting).
- **Task 2.2: Support Vector Machines (SVM):**
  1. Implement a **Linear SVM** model. Define the concepts of **Support Vectors** and **Margins**.
  2. Generalize to the **Soft Margin SVM** for the linear but nonseparable case, explaining the role of the regularization parameter C.
  3. Implement a **Kernel SVM** (e.g., RBF kernel) for the nonlinear case. Explain how the kernel function implicitly maps data to a higher-dimensional space.
- **Task 2.3: Decision Boundaries:** Select two contrasting models (e.g., LogReg and Decision Tree) and visualize their **decision boundaries** on a 2D projection of the data. Explain the observed differences.

1.3 Phase III: Robustness, Evaluation, and MLOps

- **Task 3.1: Model Evaluation and Imbalance:**
  1. Evaluate the performance of all implemented models using standard metrics (Accuracy, F1-score).
  2. Given that medical datasets are often imbalanced, specifically evaluate models using **Precision-Recall** metrics and plot the **Precision-Recall Curve (PRC)**. Explain why PRC is preferred over ROC for imbalanced datasets.
- **Task 3.2: Robustness against Label Uncertainty:**
  1. Simulate a scenario of **uncertain labels** by intentionally flipping 5% of the training labels for the minority class.
  2. Retrain the three best-performing models (e.g., Boosted Tree, Kernel SVM, LogReg with regularization).
  3. Compare the drop in performance between the baseline models and the robust models (using higher regularization in LogReg/SVM, or depth control in Trees).
- **Task 3.3: MLOps with MLflow:**
  1. Configure an **MLflow Experiment**.
  2. Log the hyperparameters (e.g., K for KNN, C for SVM, depth for Tree) and the evaluation metrics (Precision, Recall, F1) for the training runs in Task 3.2.
  3. Log the final classification model as an artifact.

1.4 Phase IV: Conceptual Link to Evidential Deep Learning (EDL)

This phase is designed to elevate the TP to a research-grade report.

- **Task 4.1: Conceptual Modeling of Uncertainty:**
  1. Read and summarize the core principles of the target paper regarding **label uncertainty** and **Evidential Deep Learning (EDL)**.
  2. Briefly explain how EDL models (like Dempster-Shafer Theory applied to neural networks) differ from traditional softmax probability models in representing *ignorance* or *uncommitted belief* in classification.
- **Task 4.2: Draft Article Structure (Rendu):** The final deliverable is a draft article based on the results of this TP. Provide a detailed outline for the paper, ensuring all sections of a top conference paper are covered.
  1. **Title:** (Suggest an appropriate title reflecting your results).
  2. **Abstract:** (150 words summarizing problem, models, and findings).
  3. **Introduction:** (State the problem, lack of robustness in classical ML to label uncertainty, and the need for evidential methods).
  4. **Related Work:** (Briefly cover standard classification, regularization, and EDL concept).

5. **Methodology:** (Detail the implementation of LogReg-SGA, Kernel SVM, and Boosting, and the robustness variant developed in Task 3.2).
6. **Experiments and Results:** (Tabulate all metrics, compare performance, and analyze the robustness against simulated label noise (Task 3.2)).
7. **Conclusion:** (Summarize the main contribution and propose future work towards implementing a full EDL model).

---

2 Code Placeholder for Logistic Regression with SGA

The code below provides a conceptual template for implementing Logistic Regression (Task 1.2).

```
Code Python 1.2: Logistic Regression with SGA (Conceptual)
import numpy as np
from sklearn.metrics import accuracy_score

# Assume X_train_scaled and y_train are available

def sigmoid(z):
    return 1 / (1 + np.exp(-z))

def stochastic_gradient_ascent(X, y, learning_rate=0.01, epochs=100):
    m, n = X.shape
    # Initialize weights (theta) and bias
    theta = np.zeros(n)
    bias = 0

    for epoch in range(epochs):
        # Iterate over each training example (Stochastic)
        for i in range(m):
            xi = X[i, :]
            yi = y[i]

            # Prediction
            z = np.dot(xi, theta) + bias
            h = sigmoid(z)

            # Error and Gradient Update (for one sample)
            error = yi - h

            # Update theta and bias
            theta = theta + learning_rate * error * xi
            bias = bias + learning_rate * error

    return theta, bias
```

```
# Prediction function using learned parameters
def predict_proba(X, theta, bias):
    z = np.dot(X, theta) + bias
    return sigmoid(z)

# Example usage (needs actual data):
# theta, bias = stochastic_gradient_ascent(X_train_scaled, y_train)
# y_pred_proba = predict_proba(X_test_scaled, theta, bias)
# y_pred = (y_pred_proba >= 0.5).astype(int)
# print(f"SGA LogReg Accuracy: {accuracy_score(y_test, y_pred):.4f}")
```