*Project 3 Briefing:*

*Classifying Protein Localization Sites with Neural Networks*

# Project objective:

After building a solid foundation with regression problems in the first two projects, this third project will introduce students to a new, essential task in supervised learning: multiclass classification. To do so, students will work with the Yeast Dataset, which contains various features extracted from protein sequences, with the goal of predicting the cellular compartment (among 10 possible categories) where each protein localizes.  
  
This project challenges students to adapt their deep learning knowledge to classification tasks, learn new loss functions, evaluation metrics, and optimization techniques, and handle model complexity and overfitting.

## Dataset:

Yeast Dataset  
Source: UCI Machine Learning Repository  
<https://archive.ics.uci.edu/dataset/110/yeast>  
  
- Samples: 1,484  
- Features: 8 real-valued protein attributes  
- Target: Class label representing the localization site (10 classes)  
  
The dataset is moderately sized, well-suited for neural network experimentation, and presents meaningful challenges due to class imbalance.

## Structure and Notebooks

This project consists of four sequential notebooks, each building upon the previous one.

### Notebook 1: Exploring the Dataset & Understanding Multiclass Classification

**Learning Objectives:**

* Understand what multiclass classification means and how it differs from binary classification.
* Analyze the yeast dataset: class distribution, feature distributions, and correlations.
* Introduce classification-specific loss functions (Cross-Entropy) and explain their importance.
* Use visualizations to clarify multiclass settings (e.g., softmax output interpretation). (explain softmax).

**Key Concepts Introduced**

* Multiclass targets
* One-hot encoding
* Cross-Entropy Loss for classification
* Softmax function and probability outputs
* Evaluation metrics: accuracy, per-class performance, confusion matrix

### Notebook 2: First Neural Network for Multiclass Classification

**Learning Goals**

* Build and train a basic MLP to perform multiclass classification.
* Understand how to adapt input/output dimensions for classification tasks.
* Implement softmax + cross-entropy loss pipeline.
* Track training accuracy and evaluate on test set.
* Introducing the issue of class imbalance

**Key Concepts Introduced**

* Network output = number of classes (logits)
* Use of nn.CrossEntropyLoss() in PyTorch
* Model architecture for classification
* Prediction vs. probability vs. class label
* Class imbalance

### Notebook 3: Optimization Techniques & Loss Comparisons

**Learning Goals**

* Experiment with different optimizers: GD, SGD, and mini-batch SGD.
* Explain the intuition, no details, of those optimizers.
* Compare convergence speed and performance.
* Study impact of optimizer choice on classification accuracy.

**Key Concepts Introduced**

* Optimization algorithms for deep learning
* Trade-offs between batch size and gradient variance
* Loss surface intuition

### Notebook 4: Combating Overfitting & Final Evaluation

**Learning Goals**

* Analyze overfitting in classification networks.
* Implement regularization techniques such as **Dropout and L2 regularization**.
* Compare training vs. validation performance.
* Select and justify the best model.
* Use precision, recall, and F1 scores for more nuanced evaluation.

**Key Concepts Introduced**

* Dropout layers (nn.Dropout in PyTorch)
* Weight decay in optimizers
* Early stopping logic (conceptual)
* Generalization vs. memorization

## Outcomes

By the end of this project, students will:  
- Understand and implement multiclass classification models with PyTorch.  
- Choose appropriate loss functions for classification.  
- Evaluate performance across imbalanced classes.  
- Optimize training with different strategies.  
- Apply regularization to improve generalization.

**What’s Coming in the Next Project**

**Project 4: Advanced Deep Neural Networks**

* Learn how to build and train **deep networks** with more layers and more complex architecture.
* Introduce classic deep architectures such as **LeNet**, **AlexNet**, and **VGG**.
* Address challenges like the **vanishing and exploding gradient problems**, and introduce basic mitigation strategies (e.g., better initialization, batch normalization).

**Project 5: Convolutional Neural Networks (CNNs) Overview**

* Present the fundamental concepts of **Convolutional Neural Networks (CNNs)**: convolutional layers, pooling layers, and fully connected layers.
* Discuss **real-world applications** of CNNs, particularly in image classification.
* Revisit foundational CNN architectures such as **LeNet**, **AlexNet**, and **VGG** from a structural point of view.

**Project 6: Advanced CNN Architectures**

* Introduce **Transfer Learning** and how pre-trained models can be adapted to new tasks.
* Explain **Data Augmentation** as a technique to improve model generalization.
* Explore advanced architectures such as **ResNet** and **Inception Networks**, focusing on their structural innovations and training efficiency.