# Intelligence System:

# Semi-supervised Learning in Machine Learning

សាកលវិទ្យាល័យន័រតុន
NORTON UNIVERSITY

Intelligence System Development

2024 – 2025
Y4E1 – DCS – NU

**By: SEK SOCHEAT**
Advisor to DCS and Lecturer
**Mobile:** 017 879 967
**Email:** socheat.sek@gmail.com

# Table of Contents

# Introduction to Semi-supervised Learning

Semi-supervised learning (SSL) is a machine learning technique that bridges the gap between supervised and unsupervised learning.

It is particularly useful when there is a small amount of labeled data and a large amount of unlabeled data. SSL leverages both to improve the performance of a predictive model.

The core idea is to use the structure of the unlabeled data to complement the information in the labeled data.

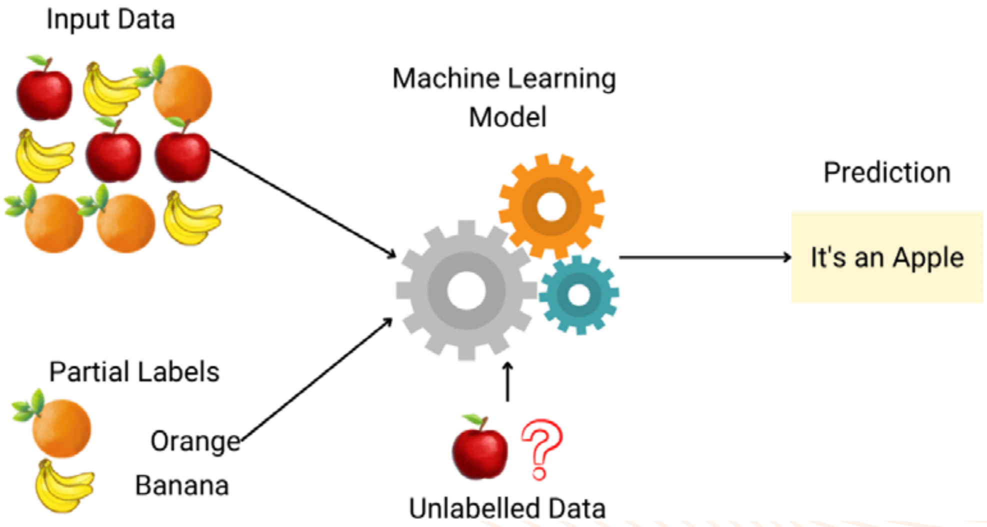



Image from google

# Key Characteristics of Semi-Supervised Learning

Semi-supervised learning combines a small amount of labeled data with a large amount of unlabeled data to improve learning accuracy.

*Its key characteristics include:*

**1. Data Distribution:** Works with labeled and unlabeled data.

**2. Cost Efficiency:** Reduces the dependency on labeled data, which is often expensive or time-consuming to obtain.

**3. Learning Paradigm:** Improves generalization by learning from the underlying data distribution.

**4. Assumptions:** Posits that high-dimensional data lies on a simpler, lower-dimensional structure, making it easier to identify relationships and assign labels.

- **Continuity Assumption:** Nearby points likely share the same label.
- **Cluster Assumption:** Points in the same cluster typically have the same label.
- **Manifold Assumption:** High-dimensional data resides on a simpler, lower-dimensional structure, aiding labeling.

# Types of Semi-Supervised Learning

Semi-supervised learning employs various methods to leverage both labeled and unlabeled data effectively.

***Key approaches include:***

- **Self-Training:** A model trained on labeled data predicts labels for unlabeled data, which are added back to training iteratively.
- **Co-Training:** Two models with different feature sets label each other's data, enhancing learning.
- **Generative Models:** Assumes the data comes from a joint distribution of features and labels, enabling inferences from unlabeled data.
- **Graph-Based Learning:** Treats data points as nodes in a graph, with edges representing similarity, propagating labels across the graph.

# Popular Algorithms in Semi-Supervised Learning

A variety of algorithms have been developed to utilize labeled and unlabeled data.

***Prominent examples include:***

- **Self-Training:** Train on labeled data, predict labels for unlabeled data, and iteratively add confident predictions to refine the model.
- **Co-Training:** Use multiple models with different feature sets to label data for each other, improving robustness.
- **Generative Models:** Variational Autoencoders (VAEs) and Gaussian Mixture Models (GMMs) model data generation and infer labels for unlabeled data based on the modeled distributions.
- **Graph-Based Methods:** Represent data points as nodes and similarity as edges; propagate labels through the graph to label unlabeled data.
- **Deep Learning:** Semi-Supervised Generative Adversarial Networks (GANs) and Ladder Networks extract deep features and patterns from labeled and unlabeled data to achieve high accuracy.

# Steps in Semi-Supervised Learning

The process of semi-supervised learning typically involves the following steps:

**1. Data Collection:** Gather both labeled and unlabeled datasets, ensuring sufficient representation for effective learning.

**2. Data Preprocessing:** Clean, normalize, and format the data, maintaining consistency between labeled and unlabeled examples to ensure compatibility.

**3. Model Initialization:** Begin by training the model using only the labeled data to establish a baseline performance.

**4. Incorporate Unlabeled Data:** Apply methods like self-training, co-training, or other semi-supervised techniques to make use of the unlabeled data.

**5. Iteration:** Refine the model iteratively by incorporating newly labeled data derived from the unlabeled set in each cycle.

**6. Validation and Testing:** Assess the model's performance using a separate validation and test set to ensure generalization and accuracy.

# Applications of Semi-Supervised Learning

Semi-supervised learning is widely applied in domains where labeled data is scarce but unlabeled data is abundant. Key applications include:

**1. Natural Language Processing (NLP):** Tasks like sentiment analysis, text classification, and machine translation benefit from unlabeled text data to enhance performance.

**2. Computer Vision:** Used for image classification and object detection, leveraging large datasets of unlabeled images alongside limited labeled examples.

**3. Healthcare:** Helps in medical diagnostics by training models with limited annotated (take notes) medical data and a vast pool of unlabeled medical records.

**4. Speech Recognition:** Combines labeled (transcribed) and unlabeled (untranscribed) audio to improve recognition accuracy.

**5. Fraud Detection:** Identifies anomalies and suspicious behavior in financial systems with a small set of labeled fraud cases and extensive transaction data.

# Challenges in Semi-Supervised Learning

Semi-supervised learning faces several challenges that can affect its effectiveness, including:

**1. Assumption Dependency:** Relies heavily on assumptions like smoothness, cluster, or manifold assumptions, which may not hold true for all datasets.

**2. Label Noise:** Errors in predicted labels for unlabeled data can propagate and degrade model performance.

**3. Imbalanced Classes:** Class imbalance in datasets can lead to biased predictions and reduced accuracy.

**4. Model Complexity:** Some methods, such as graph-based approaches, are computationally intensive and require significant resources.

# Example of Semi-Supervised Learning

**Task:** Classify emails as spam or non-spam

- **Labeled Data:** Start with 1,000 labeled emails.
- **Unlabeled Data:** Utilize an additional 10,000 unlabeled emails.
- **Approach:**
  1. Train an initial model using the labeled emails.
  2. Predict labels for the unlabeled emails.
  3. Add confidently predicted labels back to the training set.
  4. Re-train the model iteratively.

**Outcome:**

- Improved classification accuracy by effectively leveraging the unlabeled data.

NORTON UNIVERSITY

# Example of Semi-Supervised Learning

## Classify emails as spam or non-spam

```python
import numpy as np # For numerical operations
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer # To transform text into numerical features.
from sklearn.linear_model import LogisticRegression # A simple classifier for spam detection.
from sklearn.metrics import accuracy_score # To evaluate model performance.

# A small set of emails with known labels (1 for spam, 0 for not spam).
# Labeled Emails
labeled_emails = [
    "Win a free iPhone now!",                    # Spam
    "Urgent: Your account is locked.",           # Spam
    "Meeting tomorrow at 10AM",                  # Not Spam
    "Congratulations, you've won a prize!",      # Spam
    "Lunch at 12?",                              # Not Spam
    "Exclusive offer just for you!",             # Spam
    "Don't forget to submit the report.",        # Not Spam
    "Your payment is due immediately.",          # Spam
    "See you at the conference next week.",      # Not Spam
    "Important security update required."        # Spam
]
labeled_labels = [1, 1, 0, 1, 0, 1, 0, 1, 0, 1]  # 1 = Spam, 0 = Not Spam

# Emails without labels, simulating real-world unlabeled data.
unlabeled_emails = [
    "Hurry! Sale ends tonight.",
    "Project deadline extended to next Friday.",
    "Your loan application is approved!",
    "Let's grab dinner this weekend?",
    "You have been selected for a cash reward.",
    "Looking forward to your feedback on the proposal.",
    "Access your account now to avoid deactivation.",
    "Are you attending the workshop next week?",
    "Get a 50% discount on all products today.",
    "Thank you for your recent purchase."
]

# Emails reserved for evaluating the model.
test_emails = [
    "Claim your free gift card now!",            # Spam
    "Let's catch up this weekend.",              # Not Spam
    "Your order has been shipped.",              # Not Spam
    "Congratulations! You've been chosen.",      # Spam
    "Complete your payment to avoid cancellation." # Spam
]
test_labels = [1, 0, 0, 1, 1]  # 1 = Spam, 0 = Not Spam
```

```python

# Convert text data into numerical feature matrices
vectorizer = CountVectorizer() # CountVectorizer: Transforms text into a bag-of-words representation
X_labeled = vectorizer.fit_transform(labeled_emails) # fit_transform: Learns the vocabulary from labeled emails and transforms them.
X_unlabeled = vectorizer.transform(unlabeled_emails) # transform: Applies the learned vocabulary to unlabeled and test emails.
X_test = vectorizer.transform(test_emails)

# Train the first model using only the labeled data.
model = LogisticRegression() # LogisticRegression: A simple supervised learning model.
model.fit(X_labeled, labeled_labels) # fit: Fits the model to the labeled dataset.

# Predict labels for the unlabeled emails and select high-confidence predictions.
unlabeled_predictions = model.predict_proba(X_unlabeled) # predict_proba: Returns probabilities for each class (spam or not spam).
# confidence_threshold: A minimum confidence level (80% here) for including predictions in the training data.
confidence_threshold = 0.8  # Only add high-confidence predictions
# np.where: Identifies indices of predictions exceeding the confidence threshold.
high_confidence_indices = np.where(np.max(unlabeled_predictions, axis=1) > confidence_threshold)[0]
# np.argmax: Retrieves the predicted class (label) for these high-confidence predictions.
high_confidence_labels = np.argmax(unlabeled_predictions[high_confidence_indices], axis=1)

"""
- Add confident predictions to the training set.
- Extend the labeled dataset with confident predictions.
"""
# np.vstack: Combines labeled and selected unlabeled data into one training feature matrix.
X_labeled_extended = np.vstack([X_labeled.toarray(), X_unlabeled[high_confidence_indices].toarray()])
# np.hstack: Appends high-confidence labels to the existing label set.
y_labeled_extended = np.hstack([labeled_labels, high_confidence_labels])

# Re-train the model and Incorporates both the original labeled data and the confident predictions from unlabeled data.
model.fit(X_labeled_extended, y_labeled_extended)

# Evaluate on test data
test_predictions = model.predict(X_test) # predict: Predicts labels for the test emails.
accuracy = accuracy_score(test_labels, test_predictions) # accuracy_score: Calculates the proportion of correct predictions.

print("Test Accuracy:", accuracy) # Print the accuracy to evaluate the improvement.

```

# How Its Work:

1. Train a model with labeled data.
2. Predict labels for unlabeled data and select confident predictions.
3. Add confident predictions back to the training set.
4. Retrain the model iteratively with the expanded dataset.
5. Validate the performance on a separate test set.
6. This process demonstrates a practical semi-supervised learning loop.

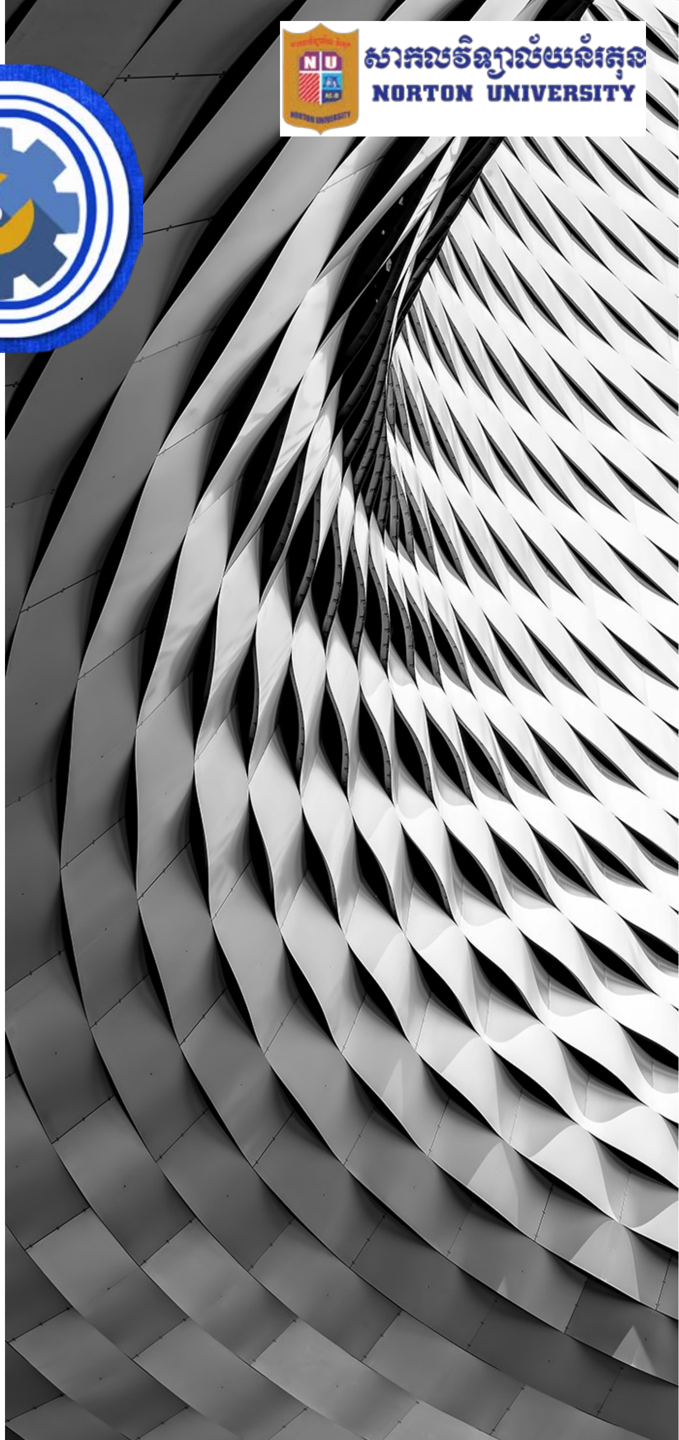
# Homework:

## Answer Questions below:


**1.** What is the primary goal of semi-supervised learning?

**2.** What are some common techniques used in semi-supervised learning?

**3.** What are the main challenges of semi-supervised learning?



# Thank you