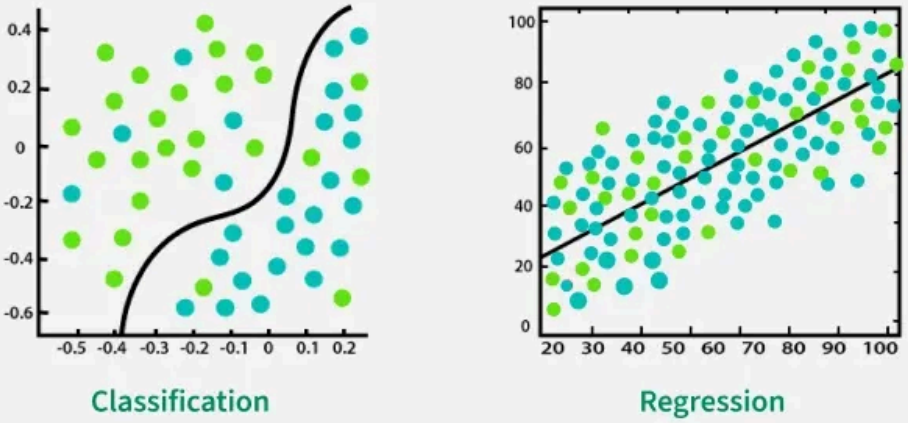
SUPERVISED MACHINE LEARNING: CLASSIFICATION

MODULE 1:

LOGISTIC REGRESSION

TABLE OF CONTENTS

Supervised Learning Overview

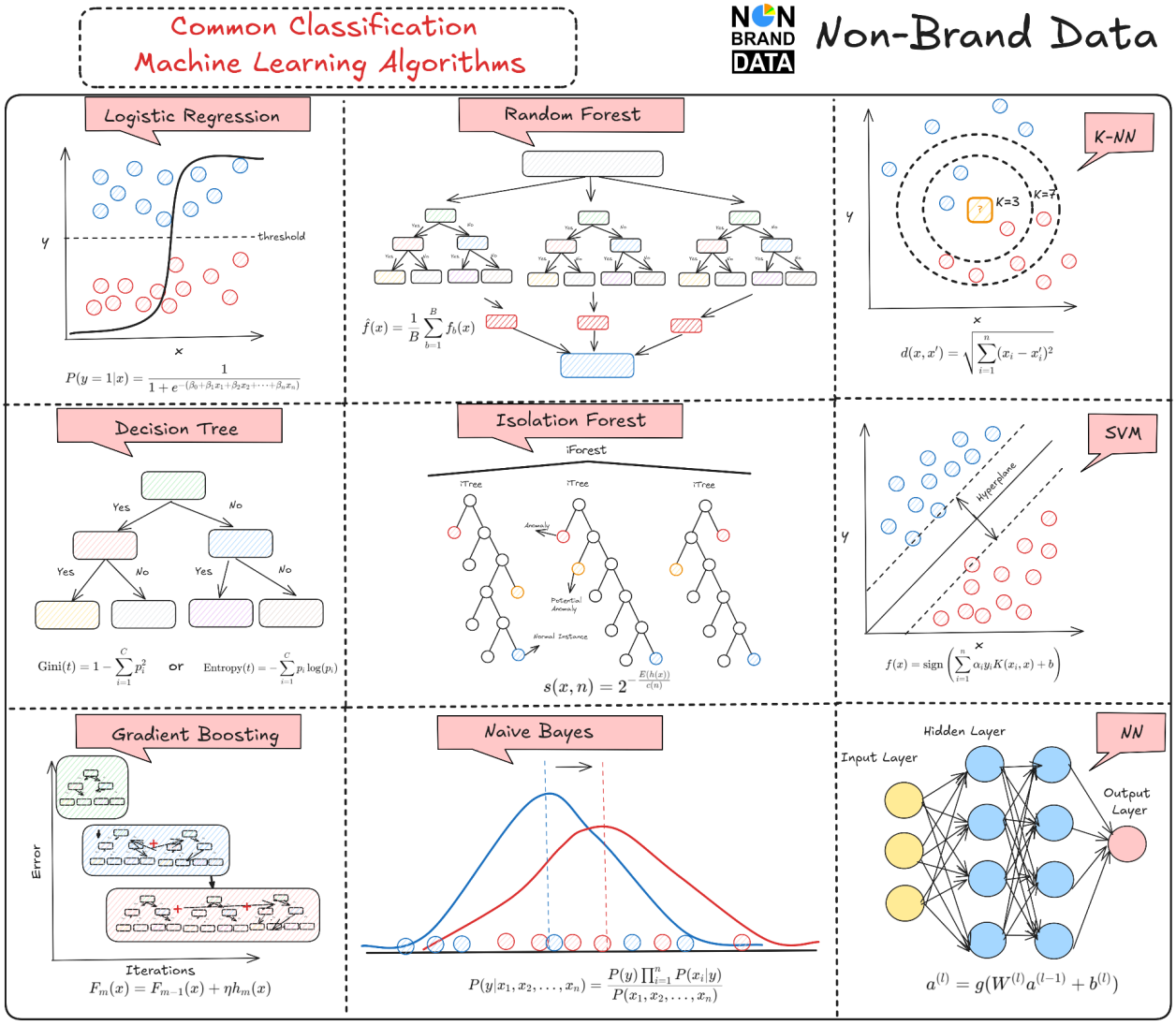


- **Two types** of supervised learning:
  - **Regression:** predicts a *continuous value* (e.g., house price).
  - **Classification:** predicts a *category or class* (e.g., fraud detection).
- Use **regression** for "how much" questions, **classification** for "which class / yes-no" problems.
  Examples:
  - Regression: house price, event attendance.
  - Classification: fraud detection, customer churn, loan default.

What's Needed for Classification

- **Feature space (không gian đặc trưng)** — numerical representation of data (e.g., petal color, shape).
- **Labeled data (dữ liệu có nhãn)** — known class for training.
- **Similarity measure (độ tương đồng)** — used to compare new samples to past examples.

Common Classification Models



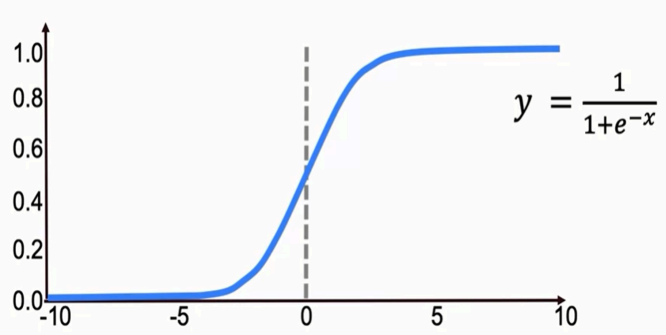
- Logistic Regression
- K-Nearest Neighbors (KNN)
- Support Vector Machines (SVM)
- Decision Trees
- Neural Networks
- Random Forests
- Boosting / Ensemble Methods

Note: All (except logistic regression) can be used for both regression and classification.

Logistic Regression

- Extends linear regression for classification.
- Models **probability** of belonging to a class using the **logistic (sigmoid)** function:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$



- Always outputs between 0 and 1.
- Decision boundary: probability = 0.5 → class 0 or 1.
- Converts linear output into **odds** and **log-odds (logit)**:

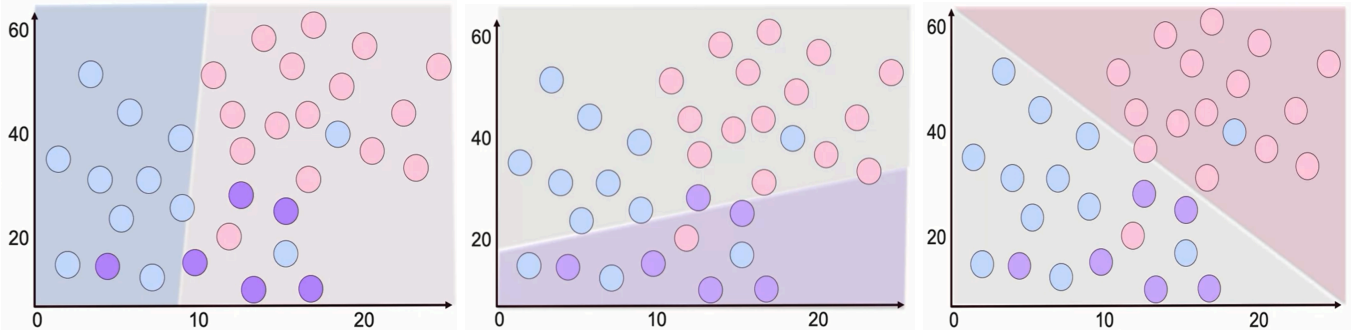
$$\text{logit}(p) = \ln\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 x$$

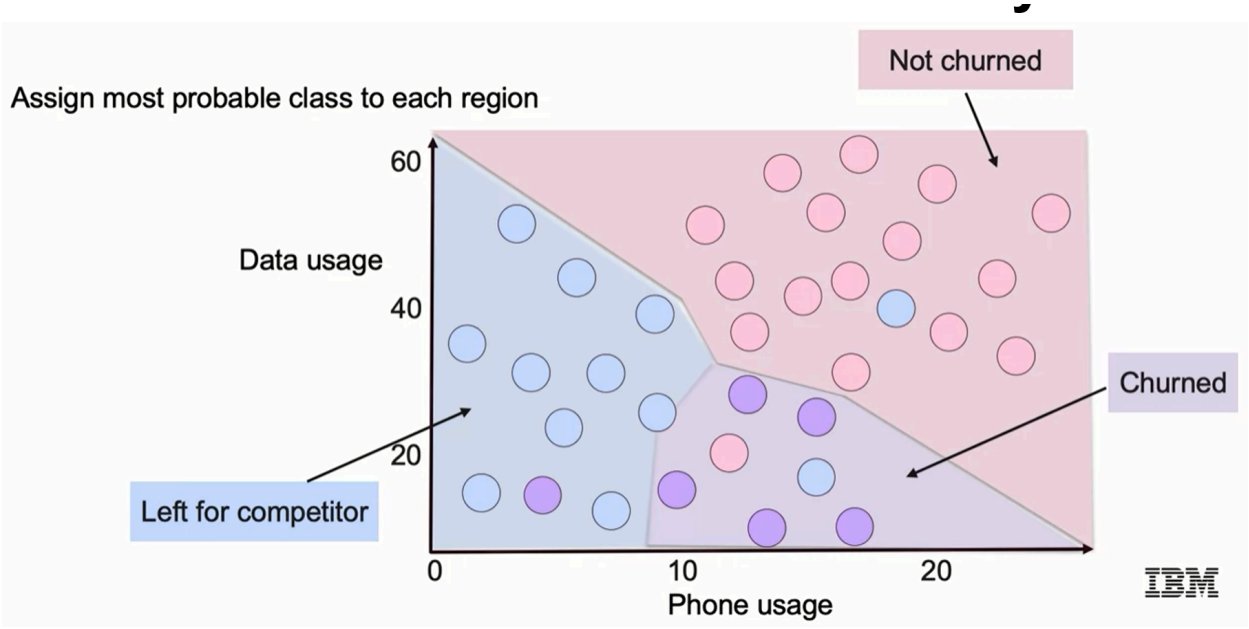
- Each unit increase in x changes log-odds by β_1.

Multi-Class Classification

Uses **One-vs-All** approach:

- Build a separate logistic model for each class vs. the rest.
- Choose the class with the highest probability.





Logistic Regression in Python (Scikit-learn)

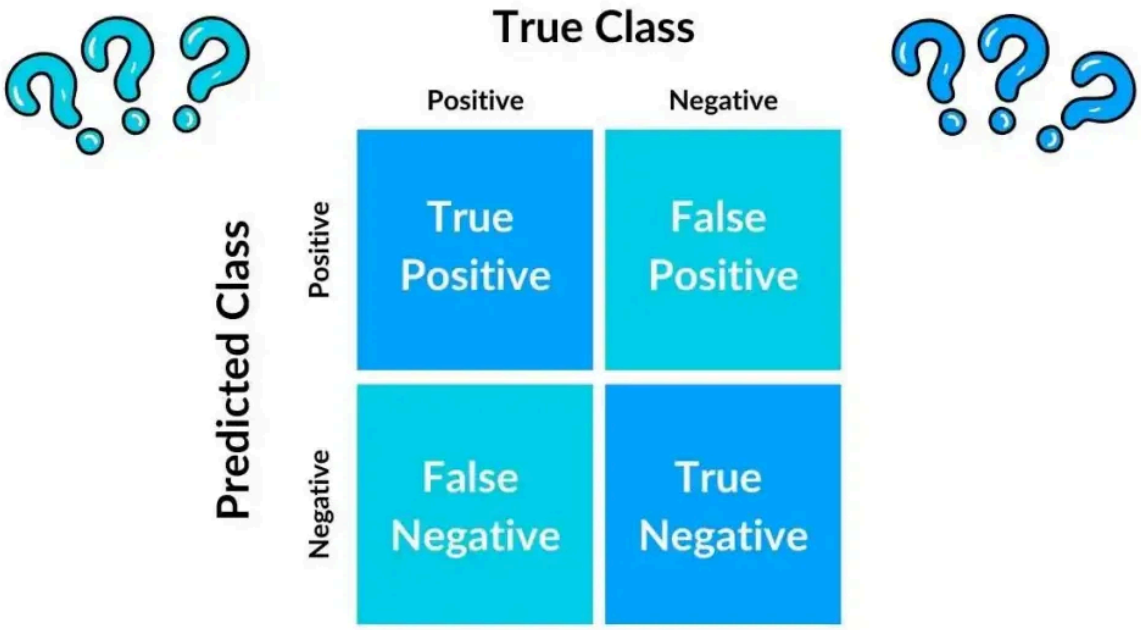
```
from sklearn.linear_model import LogisticRegression
lr = LogisticRegression(C=1.0, penalty='l2')
lr.fit(X_train, y_train)
pred = lr.predict(X_test)
lr.coef_
```

- **C** is the inverse of regularization strength (λ).
- For statistical inference (e.g., p-values), use `statsmodels`.
- For parameter tuning, use `GridSearchCV` or `cross-validation`.

Real-World Applications

- Predicting **customer churn**, **fraud detection**, **loan default**, or **top spenders**.
- Useful for both **prediction** and **interpretation** — coefficients show how features influence outcome (tác động của đặc trưng đến kết quả).

Confusion Matrix

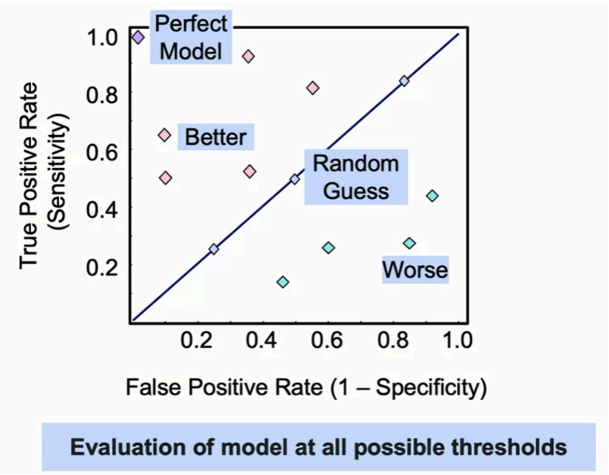


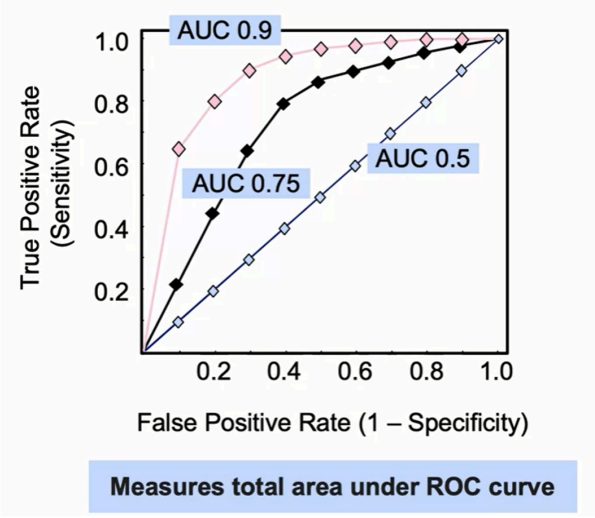
Key Metrics

Metric	Formula	Meaning
Accuracy	(TP + TN) / Total	Overall correctness
Precision	TP / (TP + FP)	How often “positive” predictions are correct
Recall (Sensitivity)	TP / (TP + FN)	How many actual positives were caught

Specificity	TN / (TN + FP)	How well negatives are identified
F1 Score	2 × (Precision × Recall) / (Precision + Recall)	Balances precision and recall

ROC & Precision–Recall Curves



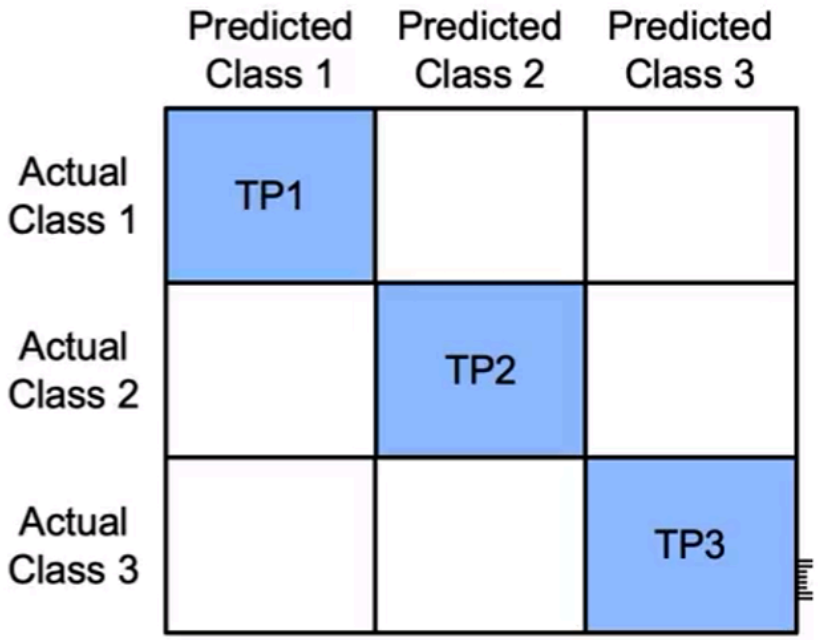


- **ROC (Receiver Operating Characteristic)** plots:
  - y-axis: True Positive Rate (Recall)
  - x-axis: False Positive Rate (1 – Specificity)
- **AUC (Area Under Curve):** higher = better model.
- **Precision–Recall Curve:**
  - Better for **imbalanced datasets**.
  - Shows trade-off between detecting positives and avoiding false alarms.

Rule of thumb:

- Balanced classes → use **ROC AUC**.
- Imbalanced classes → use **Precision–Recall**.

Multi-Class Metrics



- Extend confusion matrix to $n \times n$.
- Compute **precision, recall, F1** for each class (one-vs-all).
- Evaluate based on **misclassification cost** — which class matters more.

```
from sklearn.metrics import accuracy_score,
precision_score, recall_score, f1_score,
roc_auc_score, confusion_matrix
```

Key Takeaways

- **Regression → continuous values; Classification → discrete classes.**
- **Logistic Regression** uses **sigmoid** to map predictions to probabilities.
- **One-vs-All** extends binary logistic regression to multi-class problems.
- **Choosing the right metric** (accuracy, recall, precision, F1, ROC, etc.) depends on the **business goal** and **data balance**.
- **Scikit-learn** provides efficient tools for model fitting and evaluation.