

- title: "Intro to Machine Learning"summary: Machine Learning is the study of making machines learn a concept without explicitly programming it. It involves building algorithms that can learn from input data to make predictions or find patterns in the data.date: 2022-10-11categories: supervised-learning | Bayes classifier
    - What is Machine Learning?
    - Why Do We Need Machine Learning?
    - Real-World Applications of Machine Learning
    - How Do We Approach Machine Learning?
  - Supervised Learning vs. Unsupervised Learning
    - Statistical Modeling Approach:
    - Classification
      - Generative Classifier
      - Discriminative Classifier
      - Maximum Likelihood Estimation (MLE)
      - Case Study: Email Classification with Bayes Classifier
        - Data
        - Step 1: Bayes Classifier
        - Step 2: Estimating Probabilities Using MLE
        - Step 3: Applying Bayes Classifier with MLE Estimates
- 

**title: "Intro to Machine Learning"**  
**summary: Machine Learning is the study of making machines learn a concept without explicitly programming it. It involves building algorithms that can learn from input data to make predictions or find patterns in the data.**  
**date: 2022-10-11 categories:**  
**supervised-learning | Bayes classifier**

---

**What is Machine Learning?**

Machine Learning is the study of making machines learn a concept without explicitly programming it. It involves building algorithms that can learn from input data to make predictions or find patterns in the data.

## Why Do We Need Machine Learning?

There are tasks that humans can distinguish but can't easily hard-code into a program.

For example, writing a set of rules to code a cat.

Sometimes, we don't even know the exact task we want to solve and just want to discover patterns in the data.

## Real-World Applications of Machine Learning

- **Recommendation Systems:** Netflix, Amazon, Overstock
- **Stock Prediction:** Goldman Sachs, Morgan Stanley
- **Risk Analysis:** Credit card companies, Insurance firms
- **Face and Object Recognition:** Cameras, Facebook, Microsoft
- **Speech Recognition:** Siri, Cortana, Alexa, Dragon
- **Search Engines and Content Filtering:** Google, Yahoo, Bing

## How Do We Approach Machine Learning?

Study a prediction problem in an **abstract manner** and come up with a solution which is applicable to many problems simultaneously.

Different types of **paradigms and algorithms** that have been successful in prediction tasks.

How to **systematically analyze** how good an algorithm is for a prediction task.

### Types of Paradigms:

- **Supervised Learning:** e.g., decision trees, neural networks
- **Unsupervised Learning:** e.g., k-means clustering, principal component analysis
- **Reinforcement Learning:** e.g., Q-learning, policy gradients

## Evaluating Algorithms:

- Use metrics like accuracy, precision, recall, F1 score, AUC-ROC.
  - Techniques like cross-validation.
  - Consider computational complexity, interpretability, and scalability.
- 

# Supervised Learning vs. Unsupervised Learning

---

- **Data:**

$$(x_1, y_1), (x_2, y_2), \dots \in X \times Y$$

- **Assumption:** There is a (relatively simple) function:

$$f^* : X \rightarrow Y$$

such that

$$f^*(\vec{x}_i) = y_i$$

for most  $i$

- **Learning Task:** Given  $n$  examples, find an approximation:

$$\hat{f} \approx f^*$$

- **Goal:** It gives mostly correct predictions on unseen examples.

-**Data:**

$$(x_1, y_1), (x_2, y_2), \dots \in X \times Y$$

- **Assumption:** There is a (relatively simple) function:

$$f^* : X \rightarrow Y$$

such that

$$f^*(\vec{x}_i) = y_i$$

for most  $i$

- **Learning Task:** Given  $n$  examples, find an approximation:

$$\hat{f} \approx f^*$$

- **Goal:** It gives mostly correct predictions on unseen examples.

## Statistical Modeling Approach:

- **Labeled Training Data:**

$$(x_i, y_i)$$

drawn independently from a fixed underlying distribution (i.i.d assumption).

- **Learning Algorithm:** Select

$$\hat{f}$$

from a pool of models  $\mathcal{F}$  that maximize label agreement of the training data.

- **Selection Methods:** Maximum likelihood, maximum a posteriori, optimization of 'loss' criterion.

## Classification

In classification tasks, the task is to build a function that takes in a vector of **features**

$$\mathbf{X}$$

(also called "inputs") and predicts a

**label**

$\mathbf{Y}$  (also called the "class" or "output"). Features are things you know, and the label is what your algorithm is trying to figure out; for example, the label might be a binary variable indicating whether an animal is a cat or a dog, and the features might be the length of the animal's whiskers, the animal's weight in pounds, and a binary variable indicating whether the animal's ears stick up or are droopy. Your algorithm needs to tell dogs and cats apart (

$\mathbf{Y}$

*)using only this information about weight, whiskers, and ears(*

$\mathbf{X}$ .

The classifier

- **Joint Input/Output Space:**

$$X \times Y$$

- **Data Distribution:**

$$D(X \times Y)$$

- **Classifier:**

$$f : X \rightarrow Y$$

- **Goal:** Maximize the accuracy of the classifier:

$$\text{acc}(f) := P_{(x,y)}[f(x) = y] = E_{(x,y)}[1[f(x) = y]]$$

We want a classifier

$f$  that maximizes

$\text{acc}$

---

## Generative Classifier

A model with form

$$\begin{aligned} p(x, y) \\ = p(y)p(x|y) \end{aligned}$$

is called a generative classifier, since it can be used to generate examples  $x$  from each class  $y$ .

Examples of generative classifier:

- Native Bayes Classifier
- Gaussian Mixture Model (GMM)

## Bayes classifier:

$$f(\vec{x}) = \arg \max_{y \in Y} P[Y = y | X = \vec{x}]$$

## Bayes' Theorem

Bayes' theorem connects these two concepts:

$$P(Y|X) = \frac{P(X|Y) \cdot P(Y)}{P(X)}$$

Where:

- $P(Y|X)$  is the posterior probability: This is the probability of the class  $Y$  given the observed data  $X$ . It is what we ultimately want to calculate in classification tasks, and it tells us how probable a class is given the observed features.
- $P(X|Y)$  is the likelihood: This is the probability of observing the data (features)  $X$  given a particular class  $Y$ . It's used to measure how likely the observed data is for different possible classes.
- $P(Y)$  is the prior probability of the class  $Y$ .
- $P(X)$  is the evidence (overall probability of the data)

## Discriminative Classifier

A model of form

$$p(y|x)$$

is classed a discriminative classifier. It works by modeling the decision boundary directly between classes without explicitly computing the joint or conditional probabilities. Instead of modeling how data is generated, discriminative classifiers focus on learning the relationship between the features and the labels.

$$g(x) = X \rightarrow Y$$

where

$Y$  represents the set of possible class labels, typically

{

0, 1, 2, ..., k-1

}

*for*

k classes.

Examples of discriminative classifier:

- Logistic Regression
- Support Vector Machines (SVM)
- Neural Networks

## Maximum Likelihood Estimation (MLE)

In statistics, **maximum likelihood estimation (MLE)** is a method of estimating the parameters of an assumed probability distribution, given some observed data. This is achieved by maximizing a likelihood function so that, under the assumed statistical model, the observed data is most probable.

$$(\theta|X) := P(X|\theta) = P(\vec{x}_1, \dots, \vec{x}_n|\theta) = \prod_{i=1}^n P(\vec{x}_i|\theta) = \prod_{i=1}^n p_{\theta}(\vec{x}_i)$$

$$\hat{\theta}_{\text{MLE}} = \arg \max_{\theta} L(\theta|X) = \arg \max_{\theta} P(X|\theta) = \arg \max_{\theta} \prod_{i=1}^n P(\vec{x}_i|\theta) = \arg \max_{\theta} \prod_{i=1}^n p_{\theta}(\vec{x}_i)$$

## Case Study: Email Classification with Bayes Classifier

We want to classify emails as "spam" or "not spam" based on features like the presence of certain words, email length, etc.

### Data

- **Features (X):** Presence of words (e.g., "free", "win", "click"), email length, number of links, etc.
- **Labels (Y):** "spam" or "not spam"

### Step 1: Bayes Classifier

The Bayes classifier aims to find the class

y (spam or not spam) that maximizes the posterior probability

$$P(Y = y \mid X = x)$$

$$f(x) = \arg \max_{y \in \{\text{spam}, \text{not spam}\}} P(Y = y \mid X = x)$$

Using Bayes' theorem:

$$P(Y = y \mid X = x) = \frac{P(X = x \mid Y = y) \cdot P(Y = y)}{P(X = x)}$$

For classification purposes, we can ignore  $P(X = x)$  because it is the same for both classes:

$$f(x) = \arg \max_{y \in \{\text{spam}, \text{not spam}\}} P(X = x \mid Y = y) \cdot P(Y = y)$$

## Step 2: Estimating Probabilities Using MLE

To use the Bayes classifier, we need to estimate the following probabilities from our data:

1. \*\*Prior Probability

$$P(Y = y)$$

\*\* : The probability that an email is "spam" or "not spam".

2. \*\*Likelihood

$$P(X = x \mid Y = y)$$

\*\* : The probability of observing the features  $x$  given that the email is "spam" or "not spam".

## Estimating Prior Probability

$$P(Y = \text{spam}) = \frac{\text{Number of spam emails}}{\text{Total number of emails}}$$

$$P(Y = \text{not spam}) = \frac{\text{Number of not spam emails}}{\text{Total number of emails}}$$

## Estimating Likelihood

### Using MLE

We assume that the features  $x$  (e.g., presence of words) follow a certain distribution. For simplicity, let's assume that the features are binary (presence or absence of certain



words) and follow a Bernoulli distribution.

For each feature

$$X_i$$

:

$$P(X_i = 1 \mid Y = \text{spam}) = \theta_{i,\text{spam}}$$

$$P(X_i = 0 \mid Y = \text{spam}) = 1 - \theta_{i,\text{spam}}$$

Where

$$\theta_{i,\text{spam}}$$

is the probability of the word

$i$  appearing in a spam email. We use MLE to estimate

$\theta_{i,\text{spam}}$ .

## MLE for Bernoulli Distribution

For a binary feature

$$X_i$$

:

$$L(\theta_{i,\text{spam}}) = \prod_{j=1}^n \theta_{i,\text{spam}}^{x_{ij}} (1 - \theta_{i,\text{spam}})^{1-x_{ij}}$$

Where

$$x_{ij}$$

is the value of the

$i$ -th feature for the

$j$ -th email (1 if the word is present, 0 otherwise).

The log-likelihood is:

$$\log L(\theta_{i,\text{spam}}) = \sum_{j=1}^n [x_{ij} \log \theta_{i,\text{spam}} + (1 - x_{ij}) \log(1 - \theta_{i,\text{spam}})]$$

Taking the derivative with respect to

$$\theta_{i,\text{spam}}$$

and setting it to zero:

$$\frac{\partial}{\partial \theta_{i,\text{spam}}} \log L(\theta_{i,\text{spam}}) = \sum_{j=1}^n \left[ \frac{x_{ij}}{\theta_{i,\text{spam}}} - \frac{1 - x_{ij}}{1 - \theta_{i,\text{spam}}} \right] = 0$$

Solving for

$$\theta_{i,\text{spam}}$$

:

$$\hat{\theta}_{i,\text{spam}} = \frac{\sum_{j=1}^n x_{ij}}{n}$$

This is the proportion of emails where the word \$\$\$ appears in spam emails.

### Step 3: Applying Bayes Classifier with MLE Estimates

Using the estimated probabilities:

$$f(x) = \arg \max_{y \in \{\text{spam}, \text{not spam}\}} (P(X = x \mid Y = y) \cdot P(Y = y))$$

For each class \$\$\$:

$$P(X = x \mid Y = y) = \prod_{i=1}^m \hat{\theta}_{i,y}^{x_i} (1 - \hat{\theta}_{i,y})^{1-x_i}$$

Where

$$\hat{\theta}_{i,y}$$

is the MLE estimate of the likelihood of feature

i\$ given class

y\$.