Probability Theory and Distributions for machine learning

Youssef SALMAN

Lebanese University - Faculty of Sciences
Master 1 - Computer Sciences

26 novembre 2025

Outline

Motivation and Overview

Why Probability in Machine Learning ?

- Many modern machine learning algorithms are **probabilistic**.
- We need a language to :
  - Model **uncertainty** in data.
  - Reason about **events** and their likelihood.
  - Update our beliefs when we observe new data.
- This chapter :
  - Revisits the foundations of probability theory.
  - Introduces random variables and key distributions.
  - Illustrates all concepts with **Python** simulations.

Sample Spaces and Events

Sample Space

Definition

The **sample space** Ω is the set of all possible outcomes of a random experiment.

Examples :

- Flip a coin once : $\Omega = \{H, T\}$.
- Roll a die : $\Omega = \{1, 2, 3, 4, 5, 6\}$.
- Flip a coin twice : $\Omega = \{HH, HT, TH, TT\}$.
- Draw a card from a deck : $\Omega =$ all 52 cards.
- Measure temperature : $\Omega = \{t \in \mathbb{R} \mid -50 \leq t \leq 60\}$.
- Email classification : $\Omega = \{\text{spam}, \text{not spam}\}$.

Discrete vs Continuous Sample Spaces

- **Discrete** sample space :
  - Finite or countable number of outcomes.
  - Examples : dice rolls, coin flips, number of clicks.
- **Continuous** sample space :
  - Uncountably many outcomes (intervals of real numbers).
  - Examples : temperature, time, height, position.
- Correctly defining Ω is the first step of any probability problem.

Events

Definition

An **event** is any subset $A \subseteq \Omega$.
The event **occurs** if the outcome of the experiment lies in A.

Examples (die roll, $\Omega = \{1, 2, 3, 4, 5, 6\}$) :

- $A = \{6\}$: roll a six.
- $B = \{2, 4, 6\}$: roll an even number.
- $C = \{4, 5, 6\}$: roll a value greater than 3.

Special events :

- **Certain event** : Ω (always occurs).
- **Impossible event** : $\emptyset$ (never occurs).

Combining Events : Set Operations

Since events are sets, we can use set operations :

- Union : $A \cup B$ ("A or B").
- Intersection : $A \cap B$ ("A and B").
- Complement : A^c ("not A").
- Difference : $A - B$ ("A without B").

De Morgan's laws

$$(A \cup B)^c = A^c \cap B^c, \qquad (A \cap B)^c = A^c \cup B^c.$$

Applications in ML :

- False positives, false negatives.
- Conditions like "rainy and windy", "clicked ad but did not buy", etc.

Conditional Probability and Independence

Conditional Probability

Definition

For events A and B with $P(B) > 0$,

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)}.$$

Interpretation :

- We "restrict" the sample space to B (we know B occurred).
- Among all outcomes where B happens, measure the proportion where A also happens.

Example : Email Filtering

- S : email is spam.
- O : email contains the word “offer”.

Suppose :

$$P(S) = 0.3, \quad P(O) = 0.2, \quad P(S \cap O) = 0.15.$$

Then

$$P(S \mid O) = \frac{P(S \cap O)}{P(O)} = \frac{0.15}{0.20} = 0.75.$$

Interpretation :

- Without extra info : $P(S) = 0.3$.
- Given the word “offer” : $P(S \mid O) = 0.75$.
- Evidence strongly increases our belief that the email is spam.

Example : Medical Diagnosis (Intuition)

Let

- D : patient has a disease.
- T : test result is positive.

Suppose :

$$P(D) = 0.01, \quad P(T \mid D) = 0.95, \quad P(T \mid D^c) = 0.05.$$

Ideas :

- The disease is **rare** (1% prevalence).
- The test is quite sensitive and specific but not perfect.
- We want $P(D \mid T)$: "probability of disease given a positive test".
- This is exactly a Bayes' theorem problem.

Independence of Events

Definition

Events A and B are **independent** if

$$P(A \mid B) = P(A) \quad \text{or equivalently} \quad P(A \cap B) = P(A)P(B).$$

Interpretation :

- Knowing that B occurred does not change our belief about A.
- Typical example : successive flips of a fair coin.

Examples of Independence / Dependence

Independent : two coin flips

- A : first flip is Heads.
- B : second flip is Heads.
- $P(A) = 0.5,\ P(B) = 0.5,\ P(A \cap B) = 0.25 = P(A)P(B).$

Dependent : two cards without replacement

- A : first card is an Ace.
- B : second card is an Ace.
- $P(B \mid A) = 3/51 \neq 4/52 = P(B).$

Real-life dependence :

- R : “it rains in the morning”.
- T : “traffic is heavy”.
- Usually $P(T \mid R) > P(T)$.

Why Conditional Probability & Independence Matter in ML

- **Naive Bayes** : assumes features are conditionally independent given the class.
- **Bayesian models** : rely heavily on conditional probabilities.
- **Understanding dependence** :
  - Helps interpret $P(\text{disease} \mid \text{symptom})$, $P(\text{purchase} \mid \text{ad click})$, etc.
- Evaluating whether independence assumptions are reasonable is crucial for model validity.

Bayes' Theorem

Bayes' Theorem : Formula

Bayes' Theorem

For events A and B with $P(A) > 0$,

$$P(B \mid A) = \frac{P(A \mid B)\, P(B)}{P(A)}.$$

Vocabulary :

- $P(B)$: **prior** probability.
- $P(A \mid B)$: **likelihood** of data given B.
- $P(A)$: **evidence** (normalizing constant).
- $P(B \mid A)$: **posterior** probability after observing A.

Example : Weather Prediction

Let

- R : it will rain today.
- C : the sky is cloudy in the morning.

Given :

$$P(R) = 0.3, \quad P(C) = 0.4, \quad P(C \mid R) = 0.8.$$

We want $P(R \mid C)$:

$$P(R \mid C) = \frac{P(C \mid R)P(R)}{P(C)} = \frac{0.8 \times 0.3}{0.4} = 0.6.$$

Interpretation :

- Prior belief of rain : 0.3.
- After observing clouds : posterior increases to 0.6.

Bayes' Theorem in Machine Learning

- **Bayesian inference** :
  - Model parameters are treated as random variables.
  - Priors updated into posteriors with data.
- **Naive Bayes classifier** :
  - Uses Bayes' rule to compute $P(\text{class} \mid \text{features})$.
- **Decision making under uncertainty** :
  - Robotics, NLP, recommendation systems, etc.
  - Bayes' rule combines prior knowledge and observed evidence.

Random Variables

Random Variables : Definition

Definition

A **random variable** is a function

$$X : \Omega \to \mathbb{R}$$

that assigns a real number to each outcome in the sample space.

- Usually denoted by $X, Y, Z, \ldots$
- Encode complex outcomes with simpler numerical quantities.

Example : Number of Heads in 3 Coin Flips

Experiment : flip a fair coin three times.
Sample space :

$$S = \{HHH, HHT, HTH, THH, HTT, THT, TTH, TTT\}.$$

Define

$$X = \text{number of heads.}$$

Then $X \in \{0, 1, 2, 3\}$ and :

$$P(X = 0) = \frac{1}{8}, \quad P(X = 1) = \frac{3}{8}, \quad P(X = 2) = \frac{3}{8}, \quad P(X = 3) = \frac{1}{8}.$$

This is a discrete distribution for X.

Example : Sum of Two Dice

Roll two fair six-sided dice.
Let Y be the **sum** of the two faces.

- Possible values : $Y \in \{2, 3, \ldots, 12\}$.
- Some probabilities :
  - $P(Y = 2) = 1/36$ (only $(1, 1)$).
  - $P(Y = 3) = 2/36$ ($(1, 2), (2, 1)$).
  - $P(Y = 7) = 6/36$ ($(1, 6), (2, 5), (3, 4), (4, 3), (5, 2), (6, 1)$).

Again, we obtain a discrete probability distribution for Y.

Discrete vs Continuous Random Variables

- **Discrete** random variable :
  - Takes a countable set of values (e.g. $0, 1, 2, \ldots$).
  - Examples : number of heads, number of customers, number of clicks.
- **Continuous** random variable :
  - Takes values in an interval of $\mathbb{R}$.
  - Examples : temperature, height, time to failure.

Example : Continuous Variable (Temperature)

Let Z be the midday temperature in a city during summer.

$$Z \in [20, 45].$$

We can describe Z by a **probability density function** (PDF) $f_Z(z)$.

$$P(25 \leq Z \leq 30) = \int_{25}^{30} f_Z(z)\, dz.$$

- Probability is assigned to **intervals**, not individual points.
- Continuous variables model physical quantities in many ML applications.

Mean and Variance

Mean (Expected Value)

Definition

The **mean** or **expected value** of X is

$$\mu = E[X].$$

- Discrete :
$$E[X] = \sum_i x_i \, P(X = x_i).$$
- Continuous :
$$E[X] = \int_{-\infty}^{\infty} x \, f(x) \, dx.$$
- Interpreted as the “long-run average” value.

Variance and Standard Deviation

Variance

$$Var(X) = E[(X - \mu)^2].$$

- Discrete :

$$Var(X) = \sum_i (x_i - \mu)^2 P(X = x_i).$$

- Continuous :

$$Var(X) = \int (x - \mu)^2 f(x)\, dx.$$

- Equivalent formula :

$$Var(X) = E[X^2] - (E[X])^2.$$

Standard deviation

$$\sigma = \sqrt{Var(X)}.$$

Example : Waiting Time

Waiting time X (minutes) in a coffee shop :

$$X \in \{1, 2, 3, 4, 5\}, \quad P(X = x) = \{0.1, 0.2, 0.4, 0.2, 0.1\}.$$

- Mean waiting time :
$$E[X] = 3.$$
- Variance :
$$Var(X) = 1.2.$$
- Standard deviation :
$$\sigma \approx 1.095.$$

Interpretation : most customers wait within about one minute of the average.

Probability Concepts in Python

Simulating Sample Spaces in Python

Example : fair coin toss

```
import numpy as np

# Simulate 10,000 fair coin tosses
num_tosses = 10000
tosses = np.random.choice(['H', 'T'], size=num_tosses)

# Event A: getting a Head
is_head = (tosses == 'H')
num_heads = np.sum(is_head)

prob_head_empirical = num_heads / num_tosses
print("Empirical P(H):", prob_head_empirical)
```

As n grows, empirical probability $\to$ 0.5 (Law of Large Numbers).

Conditional Probability by Simulation

Example : two coin tosses

- A : first coin is Heads.
- B : both coins show the same result.

```
num_trials = 10000
first_coin  = np.random.choice(['H', 'T'], size=num_trials)
second_coin = np.random.choice(['H', 'T'], size=num_trials)

is_first_head = (first_coin == 'H')          # A
is_same       = (first_coin == second_coin) # B

P_A_given_B = np.sum(is_first_head & is_same) / np.sum(is_same
print("Empirical P(A|B):", P_A_given_B)
```

Result ≈ 0.5 confirms independence.

Bayes' Theorem : Spam Example

```
# Given probabilities
P_S = 0.2             # prior: spam
P_W_given_S = 0.7  # "offer" in spam
P_W_given_notS = 0.1

# Derived quantities
P_notS = 1 - P_S
P_W = P_W_given_S * P_S + P_W_given_notS * P_notS

P_S_given_W = (P_W_given_S * P_S) / P_W
print("P(W)      =", P_W)
print("P(S|W)  =", P_S_given_W)
```

Shows how Bayes' theorem updates the probability of spam given a keyword.

Simulating a Discrete Random Variable

Biased coin :

$$P(X = 1) = 0.7, \quad P(X = 0) = 0.3.$$

```
outcomes = [0, 1]
probabilities = [0.3, 0.7]

num_flips = 1000
flips = np.random.choice(outcomes, size=num_flips, p=probabili

num_zeros = np.sum(flips == 0)
num_ones  = np.sum(flips == 1)

print("Empirical P(0):", num_zeros / num_flips)
print("Empirical P(1):", num_ones  / num_flips)
```

Empirical values converge to 0.3 and 0.7.

Probability Distributions

Bernoulli Distribution

Definition

A Bernoulli random variable X takes values

$$X = \begin{cases} 1 & \text{with probability } p, \\ 0 & \text{with probability } 1 - p. \end{cases}$$

- PMF :
$$P(X = k \mid p) = p^k (1-p)^{1-k}, \quad k \in \{0, 1\}.$$
- Mean : $E[X] = p$.
- Variance : $Var(X) = p(1-p)$.

Typical uses : **click** / **no click**, **spam** / **not spam**.

Binomial Distribution

Definition

$X \sim \text{Binomial}(n, p)$: number of successes in n independent Bernoulli(p) trials.

- PMF :

$$P(X = k \mid n, p) = \binom{n}{k} p^k (1-p)^{n-k}, \quad k = 0, 1, \ldots, n.$$

- Mean : $E[X] = np$.
- Variance : $Var(X) = np(1-p)$.

Example :

$$n = 5,\ p = 0.6,\ k = 3 \Rightarrow P(X = 3) \approx 0.3456.$$

Bernoulli and Binomial in Python (SciPy)

```
import numpy as np
from scipy.stats import bernoulli, binom

# Bernoulli
p = 0.7
rv_bern = bernoulli(p)
print("P(X=1):", rv_bern.pmf(1))
print("Mean:",   rv_bern.mean(), "Var:", rv_bern.var())

# Binomial
n = 10
p_bin = 0.2
rv_binom = binom(n, p_bin)
k = 3
print("P(X=3):", rv_binom.pmf(k))
print("P(X<=3):", rv_binom.cdf(k))
```

Poisson Distribution

Definition

$X \sim \text{Poisson}(\lambda)$ counts the number of events in a fixed interval when events occur independently at average rate $\lambda > 0$.

$$P(X = k) = \frac{\lambda^k e^{-\lambda}}{k!}, \quad k = 0, 1, 2, \dots$$

Properties :

- $E[X] = \lambda$.
- $Var(X) = \lambda$.

Applications :

- Number of emails per hour.
- Number of cars passing a checkpoint per minute.
- Number of defects in a production batch.

Normal (Gaussian) Distribution

Definition

A continuous random variable X follows a Normal distribution $\mathcal{N}(\mu, \sigma^2)$ if

$$f(x \mid \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right).$$

Parameters :

- μ : mean (center).
- σ^2 : variance (spread).

Many natural phenomena approximately follow a Normal distribution.

Standard Normal and Z-scores

- Standard Normal :

$$Z \sim \mathcal{N}(0, 1).$$

- Any $X \sim \mathcal{N}(\mu, \sigma^2)$ can be standardized :

$$Z = \frac{X - \mu}{\sigma}.$$

- Z-score = number of standard deviations from the mean.
- CDF of Z : $\Phi(z) = P(Z \leq z)$.

Usefulness :

- Comparing scores from different distributions.
- Computing probabilities using tables or software.

Normal Distribution in ML

- Residuals in linear regression often assumed Normal.
- Many algorithms behave better when inputs are “Gaussian-like”.
- Weight initialization in neural networks often uses Normal distributions.
- **Central Limit Theorem (CLT)** :
  - Averages of many independent variables tend to be approximately Normal.
  - Justifies the widespread use of Gaussian models.

Working with Distributions in SciPy

SciPy : Normal Distribution

```
import numpy as np
from scipy.stats import norm
import matplotlib.pyplot as plt

mu = 0
sigma = 2
my_normal = norm(loc=mu, scale=sigma)

print("PDF at x=1:", my_normal.pdf(1))
print("CDF at x=1:", my_normal.cdf(1))
print("95th percentile:", my_normal.ppf(0.95))

x = np.linspace(mu - 4*sigma, mu + 4*sigma, 200)
pdf_values = my_normal.pdf(x)

plt.plot(x, pdf_values, label="Normal PDF")
plt.xlabel("x"); plt.ylabel("Density")
plt.title("Normal Distribution (mu=0, sigma=2)")
plt.legend(); plt.show()
```

SciPy : Binomial Distribution

```
from scipy.stats import binom
import matplotlib.pyplot as plt
import numpy as np

n = 10
p = 0.5
my_binomial = binom(n=n, p=p)

print("PMF at k=5:", my_binomial.pmf(5))
print("CDF at k=5:", my_binomial.cdf(5))
print("90th percentile:", my_binomial.ppf(0.9))

k = np.arange(0, n+1)
pmf_values = my_binomial.pmf(k)

plt.stem(k, pmf_values, basefmt=" ")
plt.xlabel("Number of successes")
plt.ylabel("Probability")
plt.title("Binomial Distribution (n=10, p=0.5)")
```

Simulation and Visualization

Simulating a Binomial Distribution

```
import numpy as np
from scipy import stats
import matplotlib.pyplot as plt

n_binom = 20
p_binom = 0.6
size_binom = 1000

binomial_samples = stats.binom.rvs(
    n=n_binom, p=p_binom, size=size_binom)

k_values = np.arange(0, n_binom+1)
pmf_values = stats.binom.pmf(k_values, n=n_binom, p=p_binom)

plt.figure(figsize=(8,5))
plt.hist(binomial_samples, bins=k_values, density=True,
         alpha=0.6, label="Simulated")
plt.plot(k_values, pmf_values, "o-", label="Theoretical PMF")
plt.xlabel("Number of successes")
```

Simulating a Normal Distribution

```
from scipy import stats
import numpy as np
import matplotlib.pyplot as plt

mu_norm = 0
sigma_norm = 1
size_norm = 1000

normal_samples = stats.norm.rvs(
    loc=mu_norm, scale=sigma_norm, size=size_norm)

x_values = np.linspace(mu_norm - 4*sigma_norm,
                       mu_norm + 4*sigma_norm, 200)
pdf_values = stats.norm.pdf(x_values,
                            loc=mu_norm, scale=sigma_norm)

plt.figure(figsize=(8,5))
plt.hist(normal_samples, bins=30, density=True,
         alpha=0.6, label="Simulated")
```

Summary

Key Takeaways

- **Sample spaces** and **events** provide the basic language of probability.
- **Conditional probability** and **independence** are central for modeling relationships.
- **Bayes' theorem** allows us to update beliefs with new evidence.
- **Random variables**, their **mean** and **variance** summarize uncertainty quantitatively.
- Fundamental distributions :
  - Bernoulli, Binomial, Poisson (discrete).
  - Normal / Gaussian (continuous).
- **Python (NumPy, SciPy, Matplotlib)** lets us simulate and visualize these concepts in practice.

Next Steps

- Use these probabilistic tools in :
  - Linear and logistic regression.
  - Naive Bayes and other probabilistic classifiers.
  - Bayesian methods and advanced ML models.
- Practice :
  - Implement the code examples in a notebook.
  - Experiment with different parameters.
  - Connect theory with empirical behavior.