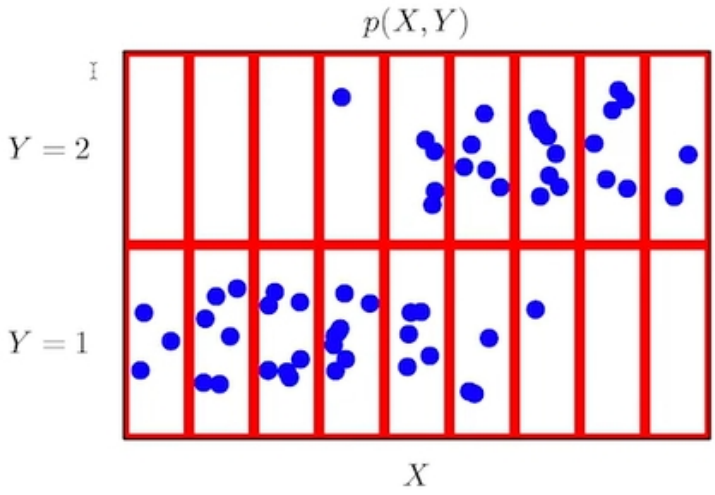
3. 확률론

3.1 확률분포는 데이터의 초상화

- $X \times Y$: 데이터 공간
- 확률분포
  - $\mathcal{D}$
  - 데이터 공간에서 데이터를 추출하는 분포
  - $\mathcal{D}$는 이론적으로 존재하는 확률분포이기 때문에 사전에 알 수 없다.
- 확률변수
  - 데이터는 확률변수로 $(\mathrm{x}, y) \sim \mathcal{D}$ 라 표기
  - $(\mathrm{x}, y) \in X \times Y$ ((x, y) : 데이터 공간 상의 **관측 가능한 데이터**)
  - 확률변수는 함수로 생각할 수 있다.



(출처: *Pattern Recognition and Machine Learning, Bishop*)

3.2 이산확률변수 vs 연속확률변수

- 확률변수는 확률분포 $\mathcal{D}$ 에 따라 **이산형(discrete)**과 **연속형(continuous)** 확률변수로 구분하게 된다.
  - 확률변수는 데이터 공간 $X \times Y$ 에 의해 결정되는 것이 아닌 확률분포 $\mathcal{D}$ 에 의해 결정된다.

3.2.1 이산형 확률변수

- 이산형 확률변수는 **확률변수가 가질 수 있는 경우의 수**를 모두 고려하여 **확률을 더해서 모델링**한다.
- $P(X = \mathrm{x})$: 확률변수가 x 값을 가질 확률 (**확률질량함수**)

$$\mathbb{P}\left(X \in A\right) = \sum_{\mathrm{x} \in A} P(X = \mathrm{x})$$

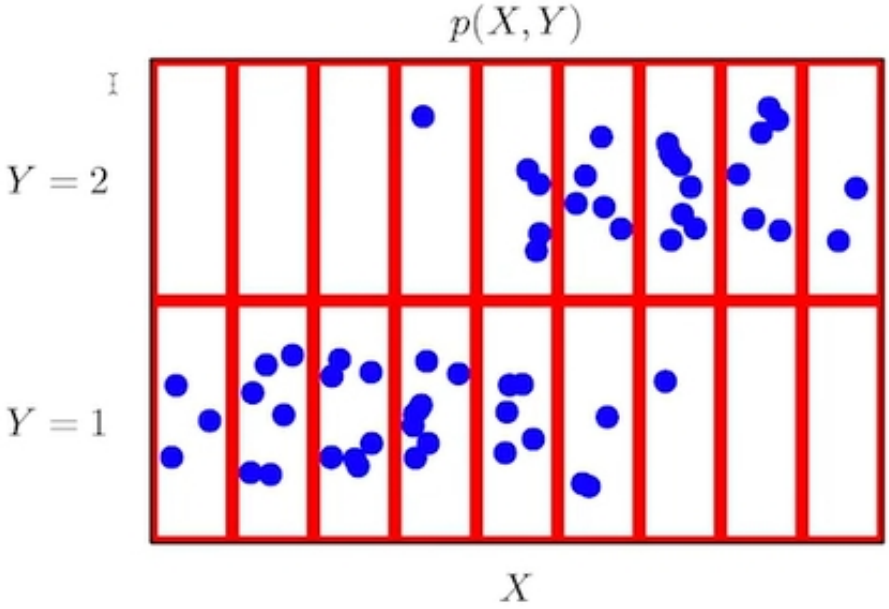
3.2.2 연속형 확률변수

- 연속형 확률변수는 **데이터 공간에 정의된 확률변수의 밀도(density)** 위에서의 **적분을 통해 모델링** 한다.
- $P(\mathrm{x})$: 누적확률분포의 변화율 (**확률밀도함수**)

$$\mathbb{P}\left(X \in A\right) = \int_A P(\mathrm{x})d\mathrm{x} = \int_A \lim_{h \to 0} \frac{\mathbb{P}\left(\mathrm{x} - h \leq X \leq \mathrm{x} + h\right)}{2h} d\mathrm{x}$$

3.3 결합분포 (joint distribution)

- 결합분포 $P(\mathrm{x}, y)$ 는 $\mathcal{D}$ 를 모델링한다.
- 주어진 데이터의 결합분포 $P(\mathrm{x}, y)$를 가지고 원래 확률분포 $\mathcal{D}$를 모델링할 수 있다.
    - 확률분포 $\mathcal{D}$ 가 이산형 확률분포일 때 결합분포 $P(\mathrm{x}, y)$는 이산형일수도, 연속형일 수도 있다.
    - 확률분포 $\mathcal{D}$ 가 연속형 확률분포일 때 결합분포 $P(\mathrm{x}, y)$는 이산형일수도, 연속형일 수도 있다.



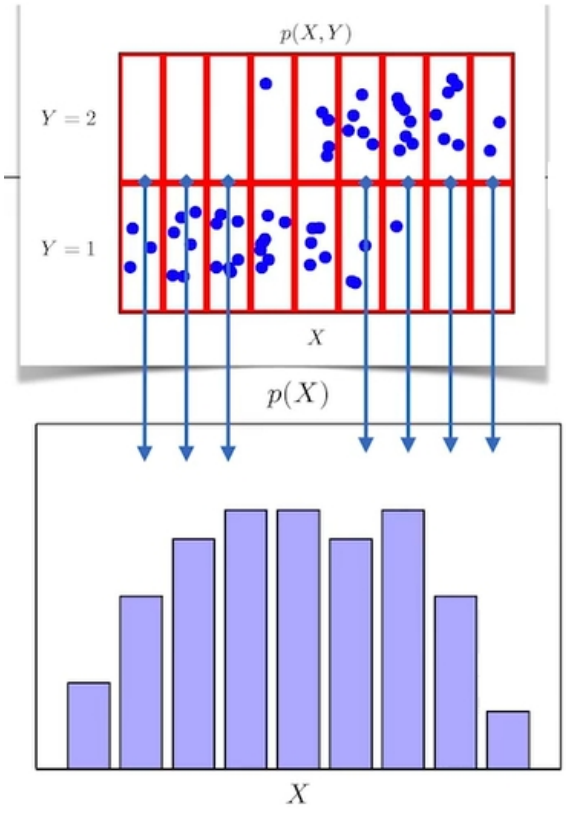
(출처: *Pattern Recognition and Machine Learning, Bishop*)

3.4 주변확률분포 (marginal distribution)

- $P(\mathrm{x})$ 는 입력 x에 대한 주변확률분포이다.
- $P(\mathrm{x})$ 는 y에 대한 정보를 주진 않는다.
- 주변확률분포 $P(\mathrm{x})$는 결합분포 $P(\mathrm{x}, y)$에서 유도 가능하다.

$$P(\mathrm{x}) = \sum_y P(\mathrm{x}, \mathrm{y}) \qquad P(\mathrm{x}) = \int_{\mathrm{y}} P(\mathrm{x}, \mathrm{y}) d\mathrm{y}$$

- x 값에 따른 빈도수 (y 상관 x)

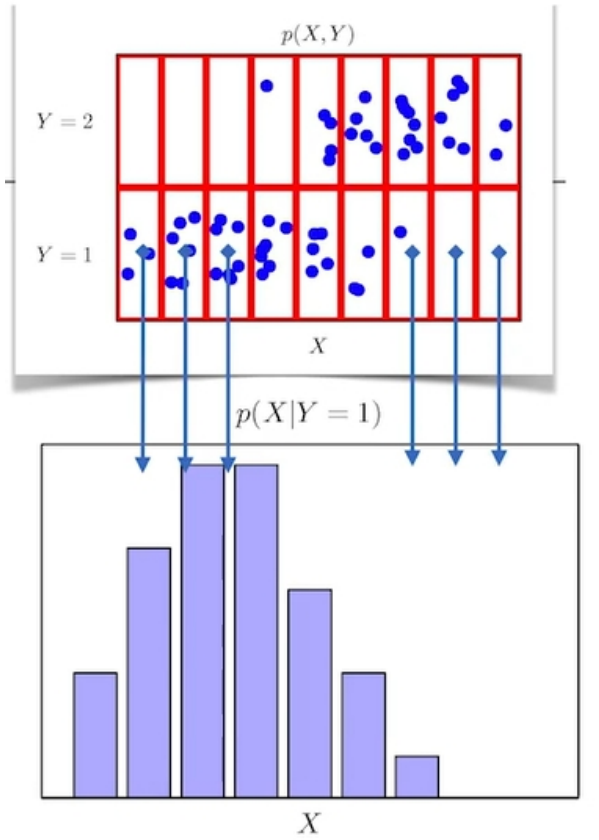


(출처: *Pattern Recognition and Machine Learning, Bishop*)

3.5 조건부확률분포

- 조건부확률분포 $P(\mathrm{x}\,|\,y)$ 는 데이터 공간에서 입력 x와 출력 y 사이의 관계를 모델링한다.
- $P(\mathrm{x}\,|\,y)$ 는 특정 클래스가 주어진 조건에서 데이터의 확률분포를 보여준다.

- y가 1일 때의 x 값에 따른 빈도수



(출처: *Pattern Recognition and Machine Learning, Bishop*)

3.6 조건부확률과 기계학습

- 조건부확률 $P(y\,|\,\mathrm{x})$ 는 입력변수 x 에 대해 정답이 y 일 확률을 의미한다.
  - 연속확률분포의 경우 $P(y\,|\,\mathrm{x})$ 는 확률이 아니도 밀도로 해석한다.

3.6.1 로지스틱 회귀

- 로지스틱 회귀에서 사용했던 선형 모델과 소프트맥스 함수의 결합은 **데이터에서 추출된 패턴을 기반으로 확률을 해석**하는 데 사용된다.

3.6.2 분류 문제

- 분류 문제에서 $\mathrm{softmax}(W\phi+\mathrm{b})$ 은 데이터 x 로부터 추출된 특징패턴 $\phi(\mathrm{x})$ 과 가중치 행렬 W 을 통해 **조건부 확률** $P(y\,|\,\mathrm{x})$ 을 계산한다.
  - $P(y\,|\,\phi(\mathrm{x}))$ 이라 써도 된다.

3.6.3 회귀 문제

- 회귀 문제의 경우 **조건부기대값** $\mathbb{E}[y\,|\,\mathrm{x}]$ 을 추정한다.
- $\mathbb{E}[y\,|\,\mathrm{x}]=\mathbb{E}_{y\sim P(y\,|\,\mathrm{x})}[y\,|\,\mathrm{x}]=\int_y yP(y\,|\,\mathrm{x})dy$
- 조건부기대값은 $\mathbb{E}||y-f(\mathrm{x})||_2$ (L_2 노름)을 최소화하는 함수 $f(\mathrm{x})$ 와 일치한다.

- 그렇기 때문에 회귀 문제에서 조건부확률 대신 조건부기대값을 사용한다.

3.6.4 딥러닝

- 딥러닝은 다층신경망(MLP, Multi Layer Perceptron)을 사용하여 특징패턴 ϕ 을 추출한다.
- 특징패턴을 학습하기 위해 어떤 손실함수를 사용할 지는 기계학습 문제와 모델에 의해 결정된다.

3.7 기대값이란?

- 확률분포가 주어지면 데이터를 분석하는 데 사용 가능한 여러 종류의 **통계적 범함수(statistical functional)를 계산**할 수 있다.
- **기대값(expectation)**
  - 데이터를 대표하는 통계량
  - 대표적인 통계적 범함수 중 하나
  - 확률분포를 통해 다른 통계적 범함수를 계산하는 데 사용

====

- **이산확률분포**의 기대값
  - **급수** 사용
  - 각 함수에 **확률질량함수**를 곱해준다.

$$\mathbb{E}_{\mathrm{x}\sim P(\mathrm{x})}[f(\mathrm{x})] = \sum_{\mathrm{x}\in\mathcal{X}} f(\mathrm{x})P(\mathrm{x})$$

- **연속확률분포**의 기대값
  - **적분** 사용
  - 각 함수에 **확률밀도함수**를 곱해준다.

$$\mathbb{E}_{\mathrm{x}\sim P(\mathrm{x})}[f(\mathrm{x})] = \int_{\mathcal{X}} f(\mathrm{x})P(\mathrm{x})d\mathrm{x}$$

- 기대값을 이용해 분산, 첨도, 공분산 등 여러 통계량을 계산할 수 있다.
  - 분산
    - $\mathbb{V}(\mathrm{x}) = \mathbb{E}_{\mathrm{x}\sim P(\mathrm{x})}\left[(\mathrm{x}-\mathbb{E}[\mathrm{x}])^2\right]$
  - 첨도
    - $\text{Skewness}(\mathrm{x}) = \mathbb{E}\left[\left(\frac{\mathrm{x}-\mathbb{E}[\mathrm{x}]}{\sqrt{\mathbb{V}(\mathrm{x})}}\right)^3\right]$
  - 공분산
    - $\text{Cov}\left(\mathrm{x}_1, \mathrm{x}_2\right) = \mathbb{E}_{\mathrm{x}_1,\mathrm{x}_2\sim P(\mathrm{x}_1,\mathrm{x}_2)}\left[(\mathrm{x}_1-\mathbb{E}[\mathrm{x}_1])(\mathrm{x}_2-\mathbb{E}[\mathrm{x}_2])\right]$

3.8 몬테카를로 샘플링

- https://www.deeplearningbook.org/contents/monte_carlo.html
- 기계학습의 많은 문제들은 **확률분포를 명시적으로 모를 때**가 대부분이다.

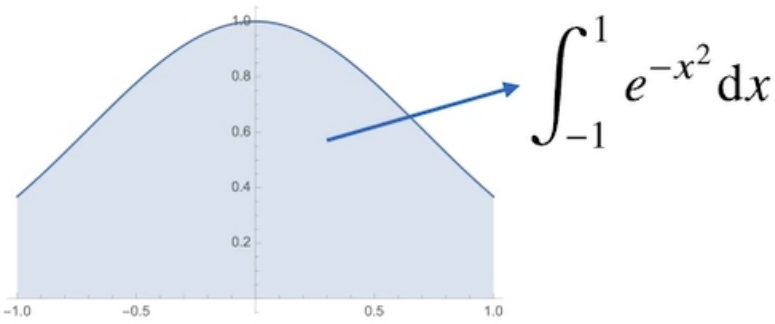
- 확률분포를 모를 때 **데이터를 이용하여 기대값을 계산**하려면 **몬테카를로(Monte Carlo) 샘플링 방법을 사용**해야 한다.
- 몬테카를로는 이산형이든 연속형이든 상관없이 성립한다.
- 확률분포에서 독립적($i.i.d.$)으로 샘플링해야 한다.

$$\mathbb{E}_{\mathrm{x}\sim P(\mathrm{x})}[f(\mathrm{x})] \approx \frac{1}{N}\sum_{i=1}^{N} f(\mathrm{x}^{(i)}), \qquad \mathrm{x}^{(i)} \overset{i.i.d.}{\sim} P(\mathrm{x})$$

- 몬테카를로 샘플링은 독립추출만 보장된다면 **대수의 법칙(law of large number)에 의해 수렴성을 보장**한다.
- 몬테카를로 샘플링은 기계학습에서 매우 다양하게 응용되는 방법이다.

3.8.1 몬테카를로 예제 : 적분 계산하기

함수 $f(x) = e^{-x^2}$ 의 $[-1, 1]$ 상에서 적분값을 어떻게 구할까?



- $f(x)$ 의 적분을 해석적으로 구하는 것은 불가능하다.
- 구간 $[-1, 1]$ 에서 균등분포를 통해 데이터를 샘플링한다.
- 구간 $[-1, 1]$ 의 길이는 2이므로 적분값을 2로 나누면 기대값을 계산하는 것과 같으므로 몬테카를로 방법을 사용할 수 있다.

$$\frac{1}{2}\int_{-1}^{1} e^{-x^2} dx \approx \frac{1}{N}\sum_{i=1}^{N} f(\mathrm{x}^{(i)}), \qquad \mathrm{x}^{(i)} \sim U(-1, 1)$$

$$\int_{-1}^{1} e^{-x^2} dx \approx \frac{2}{N}\sum_{i=1}^{N} f(\mathrm{x}^{(i)}), \qquad \mathrm{x}^{(i)} \sim U(-1, 1)$$

```
import numpy as np

def mc_int(fun, low, high, sample_size=100, repeat=10):
    int_len = np.abs(high - low)
    stat = []
    for _ in range(repeat):
        x = np.random.uniform(low=low, high=high, size=sample_size) # 균등분포에서 데이터를 샘플링
        fun_x = fun(x) # 함수 f(x) 에 값 대입
        int_val = int_len * np.mean(fun_x) # 함수값의 산술평균 x 구간의 길이(2)
        stat.append(int_val)
    return np.mean(stat), np.std(stat)

def f_x(x):
    return np.exp(-x**2)

print(mc_int(f_x, low=-1, high=1, sample_size=10000, repeat=100))
```

```
# (1.4939987699660235, 0.004011312948399921)
```

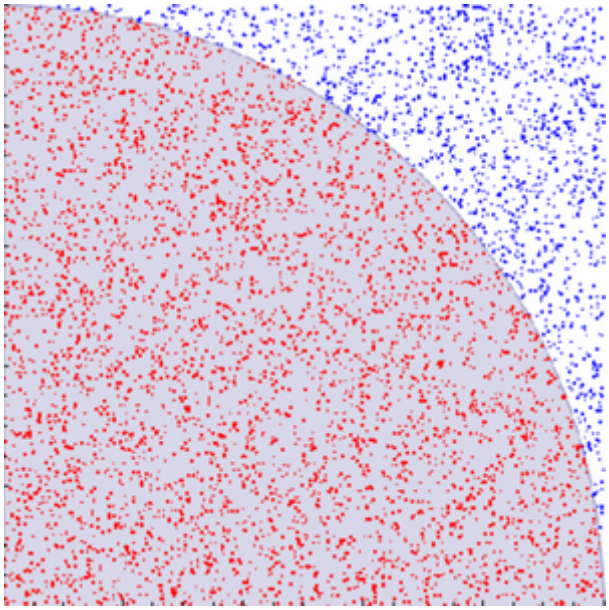
- 1.49399 ± 0.00401 이므로 오차 범위 안에 참값이 있다.

- 샘플 수가 적을 경우 몬테카를로 방법을 사용해도 오차 범위가 커질 수 있다.

3.8.2 몬테카를로 예제 : 원주율에 대한 근사값

몬테카를로 방법을 활용하여 원주율에 대한 근사값을 어떻게 구할 수 있을까?

- 원의 면적을 알면 $S = \pi r^2$ 에 의해 원주율을 알 수 있다.
- 무작위하게 수를 발생시켜 그것을 좌표로 점을 찍고, 점이 원의 영역에 포함되었는 지의 여부를 판단한다.
- 이것을 반복하여 원에 포함된 점과 그렇지 않은 점의 수를 센다.
- 두 수의 비율을 통해 원의 면적을 구한다.



```
import random

n = 1000
count = 0
for i in range(n):
    x, y = random.random(), random.random() # 0 ~ 1 사이의 난수
    if (x**2 + y**2 < 1):
        count += 1

a = 4*count/n # count/n 이 원의 1/4 에 해당하는 값이므로 4를 곱해준다.
print(a)
# 3.136
```