Day 8: Bernoulli and Binomial Distributions

Modeling Binary Outcomes and Success Counts in AI Systems

1. The Fundamental Question

How do we mathematically model binary decision outcomes and count-based success metrics that form the foundation of classification, A/B testing, and discrete event modeling in AI?

Bernoulli and Binomial distributions provide the mathematical framework for understanding binary outcomes and their aggregates, forming the statistical backbone of classification systems, hypothesis testing, and discrete event modeling in artificial intelligence.

2. Mathematical Foundations

2.1. Bernoulli Distribution: Single Binary Trial

For a random variable X representing a single binary outcome:

$$X \sim \text{Bernoulli}(p), \quad \text{where } P(X=1)=p,\ P(X=0)=1-p$$

2.2. Binomial Distribution: Multiple Independent Trials

For n independent Bernoulli trials with success probability p:

$$Y \sim \text{Binomial}(n,p), \quad P(Y=k)=\binom{n}{k}p^k(1-p)^{n-k} \text{ for } k=0,1,\dots,n$$

2.3. Key Statistical Properties

Distribution Moments and Properties

- **Bernoulli Moments:** $E[X]=p$, $\text{Var}(X)=p(1-p)$, Skewness $=\frac{1-2p}{\sqrt{p(1-p)}}$
- **Binomial Moments:** $E[Y]=np$, $\text{Var}(Y)=np(1-p)$, Skewness $=\frac{1-2p}{\sqrt{np(1-p)}}$
- **Additive Property:** If $Y_1 \sim \text{Bin}(n_1,p)$ and $Y_2 \sim \text{Bin}(n_2,p)$ are independent, then $Y_1+Y_2 \sim \text{Bin}(n_1+n_2,p)$
- **Normal Approximation:** For large n, $\text{Bin}(n,p) \approx \mathcal{N}(np, np(1-p))$

3. Comprehensive Examples and Analysis

3.1. Bernoulli Analysis: Fair Coin Toss

Single Binary Event Modeling

- $X \sim \text{Bernoulli}(p)$ with $p = 0.5$ (fair coin)
- Support: $\{0, 1\}$ where 1 represents heads, 0 represents tails
- Applications: Binary classification, decision outcomes, success/failure events

Mathematical Analysis:

$$\begin{aligned}
\text{PMF: } P(X=0) = 0.5, \quad P(X=1) &= 0.5 \\
E[X] &= 0 \cdot 0.5 + 1 \cdot 0.5 = 0.5 \\
\text{Var}(X) &= E[X^2] - (E[X])^2 = 0.5 - 0.25 = 0.25 \\
\sigma_X &= \sqrt{0.25} = 0.5
\end{aligned}$$

3.2. Binomial Analysis: Multiple Coin Tosses

Counting Successes in Repeated Trials

- $Y \sim \text{Binomial}(n = 3, p = 0.5)$ - three fair coin tosses
- Count number of heads (successes) in three independent trials
- Applications: Batch classification accuracy, A/B testing, quality control

Probability Mass Function:

$$\begin{aligned}
P(Y=0) &= \binom{3}{0}(0.5)^0(0.5)^3 = 1 \cdot 1 \cdot 0.125 = 0.125 \\
P(Y=1) &= \binom{3}{1}(0.5)^1(0.5)^2 = 3 \cdot 0.5 \cdot 0.25 = 0.375 \\
P(Y=2) &= \binom{3}{2}(0.5)^2(0.5)^1 = 3 \cdot 0.25 \cdot 0.5 = 0.375 \\
P(Y=3) &= \binom{3}{3}(0.5)^3(0.5)^0 = 1 \cdot 0.125 \cdot 1 = 0.125
\end{aligned}$$

Moment Calculations:

$$
\begin{aligned}
E[Y] &= np = 3 \times 0.5 = 1.5 \\
\text{Var}(Y) &= np(1-p) = 3 \times 0.5 \times 0.5 = 0.75 \\
\sigma_Y &= \sqrt{0.75} \approx 0.866
\end{aligned}
$$

3.3. Real-World Example: Email Spam Detection

Practical Application Scenario

- Spam detection system with 95% accuracy
- Process 1000 emails per day
- Actual spam rate: 10% of emails
- Calculate expected performance metrics

Performance Analysis:

$$
\begin{aligned}
\text{True Positives} &\sim \text{Bin}(100, 0.95) \quad E = 95,\ \text{Var} = 4.75 \\
\text{False Positives} &\sim \text{Bin}(900, 0.05) \quad E = 45,\ \text{Var} = 42.75 \\
\text{Overall Accuracy} &= \frac{95 + 855}{1000} = 95\% \\
\text{Precision} &= \frac{95}{95 + 45} \approx 67.9\%
\end{aligned}
$$

4. Why Bernoulli and Binomial are Fundamental to AI

Core Applications in Machine Learning

- **Binary Classification:** Output probabilities from logistic regression and neural networks
- **Model Evaluation:** Accuracy, precision, recall as binomial proportions
- **A/B Testing:** Statistical testing of conversion rates and success metrics
- **Bayesian Inference:** Conjugate prior relationships with Beta distribution
- **Monte Carlo Methods:** Binary event simulation and bootstrap sampling

5. Real-World AI Applications

5.1. Binary Classification Systems

- **Spam Detection:** Bernoulli output for spam/not-spam classification
- **Fraud Detection:** Binary fraud/non-fraud predictions
- **Medical Diagnosis:** Disease presence/absence classification
- **Quality Control:** Defective/non-defective product classification
- **Performance:** Modern systems achieve 95-99% accuracy on balanced datasets

5.2. Model Evaluation and Statistical Testing

- **Accuracy Testing:** Binomial tests for model performance significance
- **Cross-Validation:** Multiple train-test splits as binomial trials
- **Confidence Intervals:** Wilson score and Clopper-Pearson intervals for proportions
- **Statistical Power:** Sample size determination for A/B tests

5.3. Reinforcement Learning and Decision Making

- **Policy Decisions:** Bernoulli outcomes for action selection
- **Success Counting:** Binomial counts for reward achievement
- **Exploration-Exploitation:** Bernoulli trials for random vs. optimal actions
- **Multi-Armed Bandits:** Success counting for different strategies

6. Implementation in Modern AI Frameworks

6.1. PyTorch Implementation

```
import torch
import torch.distributions as dist
import matplotlib.pyplot as plt

# Bernoulli distribution for binary classification
p = 0.7
bernoulli = dist.Bernoulli(p)
```

```
samples = bernoulli.sample((1000,))
print(f"Empirical mean: {samples.mean():.3f} (theoretical: {p})")

# Binomial distribution for counting successes
n, p = 10, 0.3
binomial = dist.Binomial(n, p)

# PMF calculation
k_values = torch.arange(0, n+1)
pmf = torch.exp(binomial.log_prob(k_values))
print("PMF values:", pmf)

# Expected value and variance
expectation = binomial.mean
variance = binomial.variance
print(f"E[X] = {expectation:.3f}, Var(X) = {variance:.3f}")

# Application: Model accuracy analysis
def accuracy_confidence_interval(correct, total, confidence=0.95):
    """Calculate Wilson score interval for accuracy"""
    binomial_dist = dist.Binomial(total, correct/total)
    # Implementation of Wilson score interval
    z = dist.Normal(0, 1).icdf(torch.tensor((1 + confidence) / 2))
    p_hat = correct / total
    denominator = 1 + z**2 / total
    center = (p_hat + z**2 / (2 * total)) / denominator
    margin = z * torch.sqrt((p_hat * (1 - p_hat) + z**2 / (4 * total)) / total) / denomi
    return center - margin, center + margin
```

6.2. Statistical Testing and Analysis

```
import scipy.stats as stats
import numpy as np

def binomial_hypothesis_test(observed_successes, n, p_null, alternative='two-sided'):
    """Test if observed proportion differs from null hypothesis"""
    # Exact binomial test
    p_value = stats.binom_test(observed_successes, n, p_null, alternative=alternative)
    return p_value

def sample_size_power_analysis(p1, p2, power=0.8, alpha=0.05):
    """Calculate required sample size for binomial proportion test"""
    # Using normal approximation
    z_alpha = stats.norm.ppf(1 - alpha/2)
    z_beta = stats.norm.ppf(power)
    p_pool = (p1 + p2) / 2
    n = ((z_alpha * np.sqrt(2 * p_pool * (1 - p_pool)) +
          z_beta * np.sqrt(p1 * (1 - p1) + p2 * (1 - p2))) ** 2) / ((p1 - p2) ** 2)
    return int(np.ceil(n))

# Example: A/B test sample size calculation
p_control, p_treatment = 0.10, 0.12  # 10% vs 12% conversion
required_n = sample_size_power_analysis(p_control, p_treatment)
print(f"Required sample size per group: {required_n}")
```

7. Advanced Concepts and Extensions

7.1. Related Distributions

- **Multinomial Distribution:** Generalization to multiple categories
- **Negative Binomial:** Number of trials until k successes
- **Geometric Distribution:** Number of trials until first success
- **Beta-Binomial:** Overdispersed binomial data

7.2. Bayesian Perspective

- **Conjugate Prior:** Beta distribution for binomial likelihood
- **Posterior Distribution:** $P(p|data) \sim \text{Beta}(\alpha + k, \beta + n - k)$
- **Bayesian A/B Testing:** Direct probability statements about superiority
- **Thompson Sampling:** Bayesian approach to multi-armed bandits

7.3. Statistical Approximations

- **Normal Approximation:** $np > 5$ and $n(1 - p) > 5$ rule
- **Poisson Approximation:** n large, p small, np moderate
- **Continuity Correction:** Improved normal approximation for discrete variables
- **Edgeworth Expansion:** Higher-order approximation

8. Performance Analysis and Empirical Validation

Statistical Best Practices

- **Sample Size:** Minimum 30 successes and 30 failures for normal approximation
- **Confidence Intervals:** Wilson score preferred over Wald for proportions
- **Exact vs Approximate:** Use exact binomial tests for small samples
- **Overdispersion:** Check if variance exceeds $np(1-p)$ (beta-binomial may be needed)
- **Power Analysis:** Ensure sufficient sample size for hypothesis tests

9. Practical Exercises for Mastery

Hands-On Distribution Analysis

1. **Bernoulli Moments:** Derive mean, variance, and skewness for Bernoulli(p)
2. **Binomial PMF:** Implement binomial PMF calculation from first principles
3. **Normal Approximation:** Compare exact binomial probabilities with normal approximation
4. **A/B Test Design:** Calculate required sample size for a marketing experiment
5. **Bayesian Update:** Implement Bayesian updating of success probability
6. **Confidence Intervals:** Compare different methods for binomial proportion CIs

10. Common Pitfalls and Best Practices

Bernoulli and Binomial Modeling Guidelines

- **Independence Assumption:** Verify trials are independent in binomial modeling
- **Constant Probability:** Ensure success probability p remains constant across trials
- **Small Sample Correction:** Use exact methods rather than approximations for small n
- **Zero Inflation:** Check for excess zeros (zero-inflated binomial may be needed)
- **Multiple Testing:** Adjust significance levels for multiple binomial tests
- **Causality:** Remember that association in binomial tests doesn't imply causation

11. Historical Context and Modern Impact

11.1. Historical Development

- **17th Century:** Jacob Bernoulli's *Ars Conjectandi* establishes foundation
- **18th Century:** De Moivre derives normal approximation to binomial
- **19th Century:** Poisson develops Poisson approximation
- **20th Century:** Fisher and Pearson develop exact and asymptotic tests

11.2. Modern AI Applications

- **Deep Learning:** Binary cross-entropy loss for classification
- **Bayesian Neural Networks:** Bernoulli distributions for dropout and uncertainty
- **Reinforcement Learning:** Success counting for policy evaluation
- **Online Learning:** Sequential binomial testing for concept drift detection

12. Key Insight: The Foundation of Discrete Decision Making

Bernoulli and Binomial distributions provide the mathematical bedrock for understanding and modeling binary phenomena in AI systems:

- **Bernoulli as Atomic Unit:** Models individual binary decisions and outcomes
- **Binomial as Aggregate View:** Models counts and proportions across multiple trials
- **Computational Efficiency:** Simple mathematical form enables fast computation
- **Statistical Power:** Well-understood properties support rigorous inference

The elegance of these distributions lies in their ability to:

- **Scale Seamlessly:** From individual decisions (Bernoulli) to population metrics (Binomial)
- **Support Inference:** Provide exact and approximate testing methods
- **Enable Bayesian Thinking:** Natural conjugate prior relationships
- **Bridge Theory and Practice:** Simple enough for intuition, powerful enough for real applications
- **Form Building Blocks:** Foundation for more complex distributions and models

Mastering Bernoulli and Binomial distributions is essential for building AI systems that make reliable binary decisions, evaluate classification performance statistically, and understand the uncertainty in counted successes—skills that are fundamental to responsible and effective AI development.

Next: Poisson Distribution — Modeling Rare Events and Count Data

Tomorrow we'll explore the Poisson distribution and its applications in modeling count data, rare events, and arrival processes—essential for time series analysis, natural language processing, and event prediction in AI systems.