December 10, 2025

▣ **강의명:** CS109A: 데이터 과학 입문

▣ **주차:** Lecture 16

▣ **교수명:** Pavlos Protopapas, Kevin Rader, Chris Gumb

▣ **목적:** Lecture 16의 핵심 개념 학습

Contents

1 개요 **2**

2 용어 정리 **3**

3 핵심 개념 및 원리 **4**

3.1 분류(Classification)의 시작: 왜 선형 회귀는 안될까? 4

3.2 로지스틱 회귀 (Logistic Regression) 4

3.3 모델 평가와 정규화 5

3.4 베이지안 추론 (Bayesian Inference) 5

3.5 켤레 사전분포 (Beta-Binomial 모델) 6

3.6 계층 모델 (Hierarchical Models) 6

3.7 MCMC와 사후분포 샘플링 7

4 절차 및 방법 (scikit-learn 파이프라인) **8**

5 실습, 코드 및 주요 함정 **11**

5.1 주요 함정 (Gotchas) 다시보기 11

5.2 Bayesian Logistic Regression (pymc) 11

5.3 MCMC 결과 해석 (Trace Plot) 12

6 학습 체크리스트 **13**

7 FAQ (자주 묻는 질문) **14**

8 빠르게 훑어보기 (핵심 요약) **15**

1 개요

이 문서는 분류(Classification) 문제, 특히 로지스틱 회귀(Logistic Regression)의 원리를 다룹니다. 나아가 베이지안(Bayesian) 통계의 관점을 도입하여, 모델을 계층적(Hierarchical)으로 구축하고, 그 해를 MCMC(Markov Chain Monte Carlo) 시뮬레이션을 통해 추정하는 고급 기법까지 탐구합니다.

□ 핵심 요약

이 노트는 분류 모델의 기초인 로지스틱 회귀부터 시작합니다. 데이터에 비선형성을 추가하고(다항 회귀), 모델의 과적합을 방지하는 정규화(L1, L2)를 배웁니다. 이후 베이지안 관점을 도입해, 사전 확률(Prior)과 사후 확률(Posterior)의 개념을 이해하고, 켤레 사전분포(Conjugate Prior)의 편리함(베타-이항)과 한계(로지스틱 회귀)를 배웁니다. 마지막으로, 복잡한 모델(계층 모델)의 해를 구하기 위한 강력한 시뮬레이션 도구인 MCMC와 Metropolis-Hastings 알고리즘의 원리를 학습합니다.

2 용어 정리

핵심 용어들을 미리 숙지하면 뒤따르는 개념들을 이해하기 훨씬 수월합니다.

용어	쉬운 설명 (직관)	원어(영어)	비고
분류 (Classification)	데이터를 정해진 카테고리(예: 'A', 'B')로 나누는 작업.	Classification	회귀 (Regression)와 대비됨.
로지스틱 회귀	연속적인 값이 아닌, '성공/실패' 같은 확률을 예측하는 회귀.	Logistic Regression	이름은 회귀지만 분류 모델.
로그 오즈 (Log-odds)	확률(p)을 $(-\infty, +\infty)$ 범위로 변환한 값. $log(\frac{p}{1-p})$	Log-odds	로지스틱 회귀의 예측 대상.
MLE	데이터가 주어졌을 때, 이 데이터를 가장 잘 설명하는 모델 파라미터를 찾는 방법.	Max. Likelihood Est.	'이 데이터가 나올 확률이 최대가 되게 하라'
교차 엔트로피	모델의 예측 확률이 실제 정답과 얼마나 다른지(틀렸는지) 측정하는 손실 함수.	Cross-Entropy Loss	로지스틱 회귀의 손실 함수.
정규화	모델이 너무 복잡해지는 것(과적합)을 방지하기 위해 '벌칙'을 주는 기법.	Regularization	L1(Lasso), L2(Ridge) 가 있음.
사전 확률 (Prior)	데이터를 보기 전, 내가 이미 가지고 있던 믿음(지식)의 분포.	Prior Distribution	"경험상 아마 이럴 것이다."
사후 확률 (Posterior)	사전 믿음(Prior)을 데이터(Likelihood)로 업데이트한 후의 새로운 믿음.	Posterior Distribution	베이지안 추론의 최종 목표.
켤레 사전분포	사전분포와 사후분포가 같은 종류의 분포가 되는 편리한 조합.	Conjugate Prior	예: 베타(Prior) + 이항(Data) = 베타(Posterior)
계층 모델	데이터가 그룹(계층) 구조를 가질 때 (예: 학생-학교), 이를 모델링에 반영하는 기법.	Hierarchical Model	'부분적으로 정보 공유'
MCMC	사후 확률 분포가 복잡해서 수식으로 풀 수 없을 때, 샘플링(무작위 점찍기)으로 분포를 근사하는 방법.	Markov Chain Monte Carlo	베이지안 모델의 핵심 도구.

3 핵심 개념 및 원리

3.1 분류(Classification)의 시작: 왜 선형 회귀는 안될까?

'펭귄의 성별(Male/Female)'이나 '주택 구매 여부(Yes/No)'처럼, 결과가 범주형(Categorical)인 문제를 풀어야 할 때가 있습니다.

- **(1) 한 줄 요약:** 선형 회귀는 출력이 $(-\infty, +\infty)$로 뻗어 나가기 때문에, 0과 1 사이의 확률을 예측하는 분류 문제에 부적합합니다.
- **(2) 직관적 예시:** 펭귄의 부리 길이(x)로 성별(y)을 예측한다고 가정합시다. Male=1, Female=0으로 두고 선형 회귀($y = \beta_0 + \beta_1 x$)를 적용하면, 부리가 아주 긴 펭귄은 y값이 1.5, 아주 짧은 펭귄은 -0.2가 될 수 있습니다. 하지만 '확률 150%'나 '확률 -20%'는 말이 되지 않습니다.
- **(3) 기술적 설명:** 우리는 예측값이 항상 0과 1 사이에 머무르도록 강제할 필요가 있습니다. 이때 등장하는 것이 **시그모이드(Sigmoid)** 또는 **로지스틱(Logistic)** 함수입니다.

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

이 함수는 입력 z가 아무리 큰/작은 값이 들어와도, 출력은 항상 0과 1 사이의 값을 가집니다.

3.2 로지스틱 회귀 (Logistic Regression)

로지스틱 회귀는 선형 회귀의 예측값($z = \beta_0 + \beta_1 x$)을 시그모이드 함수에 통과시켜 확률로 변환하는 모델입니다.

- **(1) 한 줄 요약:** 선형 모델의 출력을 0 1 사이의 확률로 압축하여 분류 문제를 푸는 모델입니다.
- **(2) 수식의 변환 (확률, 오즈, 로그 오즈):**
  1. **확률 (Probability):** 우리가 원하는 것. $P(Y = 1|X) = p$. 범위: $[0, 1]$.

  $$p = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X)}}$$

  2. **오즈 (Odds):** 성공 확률 / 실패 확률. $Odds = \frac{p}{1-p}$. 범위: $[0, \infty]$. (확률 0.5는 오즈 1, 확률 0.8은 오즈 4)

  $$\frac{p}{1-p} = e^{\beta_0 + \beta_1 X}$$

  3. **로그 오즈 (Log-odds):** 오즈에 로그를 씌운 것. 범위: $(-\infty, +\infty)$.

  $$\ln(\frac{p}{1-p}) = \beta_0 + \beta_1 X$$

- **(3) 기술적 설명 (계수 해석):** 로지스틱 회귀의 핵심은 **로그 오즈**를 선형 예측하는 것입니다.
  β_1의 해석: X가 1단위 증가할 때, **로그 오즈**가 β_1 만큼 증가합니다.
  현실적 해석 (오즈비, Odds Ratio): X가 1단위 증가할 때, **오즈(Odds)**가 e^{β_1} **배** 증가합니다.

□ **예제:**

NBA 농구공 예시: 슛 거리(Distance)로 성공 여부(Success=1)를 예측하는 모델을 만들었습니다.

$$\ln(\frac{p}{1-p}) = 0.796 - 0.0474 \times \text{Distance}$$

- **절편 (0.796):** 거리가 0일 때(골대 바로 밑)의 로그 오즈입니다. 이때의 성공 확률 $p = \frac{1}{1+e^{-0.796}} \approx 0.689$ (약 69%).
- **계수 (-0.0474):** 거리가 1피트 증가할 때마다, 성공에 대한 로그 오즈가 0.0474만큼 감소합니다.
- **오즈비:** $e^{-0.0474} \approx 0.954$. 즉, 거리가 1피트 늘어날 때마다 성공 오즈가 약 0.954배가 됩니다 (약 4.6% 감소). 멀어질수록 슛이 어려워진다는 직관과 일치합니다.

3.3 모델 평가와 정규화

- **모델 학습 (MLE):** 로지스틱 회귀는 **최대우도추정법 (MLE, Maximum Likelihood Estimation)**으로 학습합니다. 이는 주어진 데이터가 나타날 확률을 최대화하는 β 값을 찾는 과정이며, 이는 **교차 엔트로피 손실 (Cross-Entropy Loss)**을 최소화하는 것과 수학적으로 동일합니다.
- **평가 (ROC-AUC):** 정확도(Accuracy)는 데이터가 불균형할 때(예: 90%가 Male, 10%가 Female) 모델 성능을 제대로 평가하기 어렵습니다. 이때 **ROC 커브 (Receiver Operating Characteristic Curve)**와 **AUC (Area Under the Curve)**를 사용합니다.
  - **ROC 커브:** 모든 가능한 임계값(Threshold)에 대해 '가짜 양성 비율(FPR)' 대비 '진짜 양성 비율(TPR)'을 그린 그래프입니다.
  - **AUC:** 이 커브의 아래 면적. 1에 가까울수록 모델이 양성/음성을 잘 구별한다는 의미입니다. 0.5는 무작위 추측(동전 던지기)과 같습니다.
- **정규화 (Regularization):** 모델이 너무 복잡해져 훈련 데이터에만 과적합(Overfitting)하는 것을 막기 위해, 모델의 β 계수 크기에 벌칙(Penalty)을 부과합니다.

주의사항

Scikit-Learn의 'C' 파라미터 함정

- 수학에서 정규화 강도는 λ (람다)로 표현합니다. λ가 클수록 강한 정규화입니다.
- `scikit-learn`의 `LogisticRegression(C=...)`에서 C는 $1/\lambda$ 입니다.
- 즉, **C가 작을수록 ($C = 0.01$) → 정규화가 강해지고, C가 클수록 ($C = 100$) → 정규화가 약해집니다.** 이는 직관과 반대이므로 반드시 기억해야 합니다.
- 정규화 사용 시, 모든 변수의 스케일을 맞추기 위해 `StandardScaler` 사용이 거의 필수적입니다.

3.4 베이지안 추론 (Bayesian Inference)

데이터 과학에서 모델의 파라미터(β)를 추정하는 두 가지 큰 관점이 있습니다.

- **최대우도법 (Frequentist):** "파라미터(β)는 고정된 값이다. 우리가 가진 데이터로 그 값을 가장 잘 설명하는 '단 하나의 점'을 찾자." (우리가 배운 `sklearn`의 `.fit()`이 여기에 해당)
- **베이지안 (Bayesian):** "파라미터(β)도 불확실성을 가진 '확률 분포'다. 데이터를 보기 전의 믿음(Prior)을 데이터(Likelihood)를 통해 업데이트하여, 더 정교한 믿음(Posterior)을 얻자."

$$\underbrace{P(\theta|D)}_{\text{사후 확률}} \propto \underbrace{P(D|\theta)}_{\text{우도 (Likelihood)}} \times \underbrace{P(\theta)}_{\text{사전 확률 (Prior)}}$$

3.5 켤레 사전분포 (Beta-Binomial 모델)

베이지안 계산은 복잡하지만, 특정 조합에서는 수식이 아주 깔끔하게 떨어집니다.

- **(1) 한 줄 요약:** 사전분포와 사후분포가 같은 종류의 분포가 되는 (계산이 편리한) 마법 같은 조합입니다.
- **(2) 직관적 예시 (동전 던지기):**
  - **문제:** 동전의 앞면이 나올 확률 p를 추정하고 싶다.
  - **데이터 (Likelihood):** p를 모르는 상태로 10번(n) 던져서 앞면이 7번(Σy_i) 나왔다. (이항분포/베르누이분포)
  - **사전 믿음 (Prior):** "나는 이 동전이 공정할 것 같아 ($p \approx 0.5$)." 이 믿음을 **베타 분포(Beta Distribution)**로 표현합니다. 베타 분포는 0과 1 사이 값의 불확실성을 표현하기 완벽한 분포입니다.
  - $Beta(a_0, b_0)$**의 의미:** a_0는 '사전의 가상 앞면 개수', b_0는 '사전의 가상 뒷면 개수'로 해석할 수 있습니다. "공정할 것 같다"는 $Beta(a_0 = 1, b_0 = 1)$ (모든 값 동일하게 가능) 또는 $Beta(a_0 = 5, b_0 = 5)$ (0.5 근처일 거라 강하게 믿음) 등으로 표현할 수 있습니다.
- **(3) 기술적 설명 (Posterior):** 놀랍게도, 사전분포 $Beta(a_0, b_0)$와 이항분포 데이터를 결합하면, 사후분포는 정확히

  $$Posterior \sim Beta(a_0 + \Sigma y_i, \quad b_0 + (n - \Sigma y_i))$$

  가 됩니다. (사전의 앞면 개수 + 실제 앞면 개수, 사전의 뒷면 개수 + 실제 뒷면 개수)
  사후 평균 (Posterior Mean): 이 사후분포의 평균(p의 점 추정치)은

  $$\hat{p}_{PM} = \frac{a_0 + \Sigma y_i}{a_0 + b_0 + n}$$

  이는 **사전 믿음의 평균**과 **데이터의 평균(MLE)** 사이의 **가중 평균**이 됩니다. 데이터(n)가 많아질수록 사전 믿음(a_0, b_0)의 영향력은 줄어들고 데이터의 힘이 강해집니다.

3.6 계층 모델 (Hierarchical Models)

- **(1) 한 줄 요약:** 데이터가 그룹 구조(예: 선수별 슛)를 가질 때, 각 그룹이 완전히 다르지도(No Pooling), 완전히 같지도(Complete Pooling) 않다고 보고, '부분적으로 정보를 공유'(Partial Pooling)하는 현명한 모델입니다.
- **(2) 직관적 예시 (NBA 선수 슛 예측):**
  - **문제:** 슛 거리(Distance)와 선수(Player)를 이용해 슛 성공을 예측.
  - **접근 1 (Complete Pooling):** "모든 선수는 같다." 선수를 무시하고 하나의 모델 $ln(p/(1-p)) = \beta_0 + \beta_1 \text{Dist}$ 를 씁니다. (Underfitting)
  - **접근 2 (No Pooling / OLS):** "모든 선수는 완전히 다르다." 선수를 One-Hot 인코딩하여 수백 개의 β 계수를 만듭니다.
  - **문제점:** 맥 맥클렁(Mac McClung) 선수가 2번 슛해서 2번 다 실패(0%)했다면, 이 모델은 그의 β

계수를 $-\infty$ 에 가깝게 추정하여 "그는 0

- **접근 3 (Hierarchical / Partial Pooling):** "선수들은 저마다 다르지만, 결국 모두 'NBA 선수'라는 큰 그룹에 속한다."

- **(3) 기술적 설명:** 각 선수(j)마다 고유의 절편(α_j)을 갖는다고 가정합니다.

$$\ln(\frac{p_{ij}}{1-p_{ij}}) = \alpha_j + \beta_1 \text{Distance}_{ij}$$

하지만 이 α_j 들이 완전히 따로 노는 게 아니라, **모든 선수의 평균 실력(α_0)**과 **실력의 편차(σ_α^2)**를 따르는 **정규분포**에서 추출되었다고 가정합니다.

$$\alpha_j \sim N(\alpha_0, \sigma_\alpha^2)$$

결과 (수축, Shrinkage):

- 데이터가 많은 선수(예: 스테판 커리)는 자신의 데이터로 α_j 가 결정됩니다.
- 데이터가 적은 선수(예: 맥 맥클렁)는 모델이 "데이터가 2개뿐이라 못 믿겠다"고 판단하여, α_j 를 전체 평균(α_0) 쪽으로 강하게 끌어당깁니다(수축).
- 즉, 맥 맥클렁의 0

3.7 MCMC와 사후분포 샘플링

- **문제 상황:** 베이지안 로지스틱 회귀나 계층 모델은 켤레성이 깨집니다. (사전분포 $N(\cdot)$ + 우도 $Logistic(\cdot) \rightarrow$???). 사후분포(Posterior)가 매우 복잡한 형태가 되어 수식으로 풀 수 없습니다.
- **(1) 한 줄 요약:** 사후분포라는 복잡한 산맥의 지도를 그릴 수 없을 때, 그 산맥을 무작위로 돌아다니면서(MCMC) 수천, 수만 개의 발자국(Samples)을 찍어, 그 발자국 밀도로 산맥의 형태(분포)를 역추적하는 기법입니다.
- **(2) 직관적 예시 (Rejection Sampling):**
  - **목표:** 복잡한 쿠키 모양($f(x)$)의 반죽을 찍어내고 싶다.
  - **문제:** 쿠키틀이 없지만, 그 쿠키틀을 덮는 네모난 상자($Mg(x)$)는 가지고 있다.
  - **방법:** 1. 네모난 상자($g(x)$) 안에서 무작위로 위치(x)를 하나 찍는다. 2. 그 위치(x)에서 쿠키 반죽의 높이($f(x)$)와 상자 높이($Mg(x)$)를 비교한다. 3. $f(x)/(Mg(x))$의 확률로 그 위치를 '채택(Accept)'한다. (즉, 반죽이 두꺼운 곳은 채택될 확률이 높음)
  - **결과:** 수천 번 반복하면, 채택된 점들의 분포가 정확히 쿠키 모양($f(x)$)을 따르게 됩니다.
- **(3) 기술적 설명 (Metropolis-Hastings):** Rejection Sampling은 고차원에서 매우 비효율적입니다. (거의 모든 점이 거절됨) Metropolis-Hastings는 '무작위 산책(Random Walk)'을 통해 더 효율적으로 샘플을 뽑습니다.
  알고리즘 (등산가 비유): 1. 현재 위치($\theta^{(t)}$)에 서 있습니다. (현재 위치의 높이는 $f(\theta^{(t)})$) 2. 다음 발걸음을 아무 데나 제안합니다. (θ^*) (제안 위치의 높이는 $f(\theta^*)$) 3. **비교:** 제안된 곳(θ^*)이 현재($\theta^{(t)}$)보다 **높으면 (Uphill) $\rightarrow$ 무조건 이동합니다.** 4. **비교:** 제안된 곳이 현재보다 **낮으면 (Downhill) $\rightarrow$ $R = f(\theta^*)/f(\theta^{(t)})$** (높이의 비율) 만큼의 **확률로 이동합니다.** (낮아도 가끔은 이동해야 골짜기에 갇히지 않음) 5. 이동하지 않으면, 현재 위치에 한 번 더 머무릅니다(발자국 하나 더 찍음).
  이 과정을 수천 번 반복하면, 등산가가 머물렀던 위치(발자국)의 분포가 정확히 산맥의 형태(사후분포)와 일치하게 됩니다.

4 절차 및 방법 (scikit-learn 파이프라인)

실제 데이터 분석에서는 `scikit-learn`을 사용하여 분류 모델을 효율적으로 구축합니다. 다음은 펭귄의 성별을 예측하는 전형적인 머신러닝 작업 흐름입니다.

1단계: 데이터 준비 및 전처리 (EDA)

- **데이터 로드:** 데이터를 불러옵니다. (예: 펭귄 데이터셋)
- **NaN 값 확인:** `df.isna().sum()`으로 결측치를 확인합니다.
- **NaN 값 처리:**
  - 예측 변수(Feature)의 NaN: 행 전체를 삭제(`dropna()`)하거나, 평균/최빈값 등으로 채웁니다 (`fillna()`).
  - **대상 변수(Target)의 NaN:** 해당 행은 모델 학습이나 평가에 사용할 수 없으므로, 보통 `dropna(subset=['targ`를 통해 삭제합니다.
- **대상 변수 인코딩:** 'Male', 'Female'과 같은 문자열은 모델이 이해할 수 없습니다. `LabelEncoder` 등을 사용해 0, 1과 같은 숫자로 변환합니다.

2단계: 훈련/테스트 데이터 분리

- **목적:** 모델이 처음 본 데이터(Test)에서 얼마나 잘 작동하는지 평가하기 위해 데이터를 분리합니다.
- **불균형 데이터 문제:** 만약 90%가 Male이고 10%가 Female인 데이터를 무작위로 분리하면, 테스트셋에 Female이 하나도 포함되지 않을 수 있습니다.
- **해결 (Stratified Split):** `train_test_split`의 `stratify=` 옵션에 대상 변수(y)를 지정합니다. 이는 훈련셋과 테스트셋의 0/1 비율을 원본 데이터의 비율과 동일하게 유지시킵니다.

3단계: 파이프라인 구축 및 하이퍼파라미터 튜닝 (GridSearch)

모델 학습 과정(스케일링, 모델링)을 하나로 묶고, 최적의 파라미터를 찾습니다.

주의사항

GridSearch와 K-Fold 교차 검증
모델의 성능은 데이터를 어떻게 나누었는지에 따라 우연히 좋거나 나쁘게 나올 수 있습니다. **K-Fold 교차 검증 (Cross-Validation)**은 데이터를 K개의 덩어리로 나눈 뒤, (K-1)개로 학습하고 1개로 검증하는 과정을 K번 반복하여 평균을 내는, 더 안정적인 성능 평가 방법입니다. `GridSearchCV`는 이 K-Fold 방식을 사용하여 각 하이퍼파라미터 조합의 성능을 평가합니다.

```
from sklearn.model_selection import train_test_split, KFold, GridSearchCV
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.neighbors import KNeighborsClassifier
from sklearn.pipeline import Pipeline
import numpy as np

# --- 1. 데이터준비가정 () ---
# X, y = load_penguin_data()
# y = LabelEncoder().fit_transform(y) # 'Male'/Female' -> 0/1
```

```

# --- 2. 데이터분리 ---
# X_train, X_test, y_train, y_test = train_test_split(
#     X, y, test_size=0.2, stratify=y, random_state=42
# )

# --- 3. 파이프라인및 GridSearch ---

# 3.1. 수행할작업정의스케일러 ( + 모델)
pipe = Pipeline([
    ('scaler', StandardScaler()),
    ('knn', KNeighborsClassifier())
])

# 3.2. 탐색할하이퍼파라미터그리드정의
# 모델이름'파라미터이름__' 형식
param_grid = {
    'knn__n_neighbors': [3, 5, 7, 9]
}

# 3.3. 교차검증 (CV) 방법정의
kfold = KFold(n_splits=5, shuffle=True, random_state=42)

# 3.4. GridSearch 객체생성
# cv=kfold : 5-fold 로CV 성능평가
# scoring='accuracy' : 정확도를기준으로최적모델선택
# n_jobs=-1 : 모든 CPU 코어를사용해병렬로탐색
grid_search = GridSearchCV(
    estimator=pipe,
    param_grid=param_grid,
    cv=kfold,
    scoring='accuracy',
    n_jobs=-1
)

# 3.5. 훈련데이터로탐색시작
# grid_search.fit(X_train, y_train)

# --- 4. 결과확인 ---
# print(f"Best Score: {grid_search.best_score_}")
# print(f"Best Params: {grid_search.best_params_}")
# best_model = grid_search.best_estimator_

# --- 5. 최종테스트 ---
# test_accuracy = best_model.score(X_test, y_test)
```

Listing 1: scikit-learn 파이프라인과 GridSearch 예시

4단계: 결정 경계 (Decision Boundary) 시각화

모델이 2D 공간을 어떻게 나누고 있는지 시각화하면 모델을 직관적으로 이해할 수 있습니다.

- `np.meshgrid`: 2D 평면을 촘촘한 격자(Grid)로 나눕니다. x축 좌표 배열과 y축 좌표 배열을 받아, 모든 (x, y) 좌표 쌍을 생성합니다.
- `model.predict`: 격자의 모든 점에 대해 모델이 예측(0 또는 1)을 수행합니다.
- `plt.contourf`: 예측 결과(0 또는 1)에 따라 격자점을 다른 색으로 칠하여, 모델이 만든 경계선을 시각화합니다.

5 실습, 코드 및 주요 함정

5.1 주요 함정 (Gotchas) 다시보기

주의사항

- **Pandas `value_counts()`의 함정:** `df['sex'].value_counts()`는 NaN(결측치)을 **자동으로 무시하고** 개수를 셉니다. 이로 인해 데이터가 누락되는 것을 인지하지 못할 수 있습니다. (e.g., Male 134, Female 132 = 266개. 실제 데이터 273개. 7개의 NaN이 무시됨)
- **Scikit-Learn `C` 파라미터의 함정:** $C = 1/\lambda$ 입니다. **C가 작을수록 ($C = 0.01$) 규제가 강해집니다.**
- **정규화(Regularization)와 스케일링:** L1, L2 정규화는 계수의 크기에 벌칙을 줍니다. 만약 A 변수(1000 2000)가 B 변수(0 1)보다 스케일이 훨씬 크다면, A 변수의 계수는 불공평하게 큰 벌칙을 받게 됩니다. 따라서 정규화 전 `StandardScaler`는 필수입니다.
- **계층 모델의 과적합 방지 (Mac McClung 예시):** '2번 슛, 2번 실패(0%)'라는 적은 데이터를 가진 선수에게 일반 OLS 모델은 '성공률 0%'라는 극단적인 결론을 내립니다 (과적합). 계층 모델은 이 선수의 데이터를 '평균 NBA 선수' 데이터와 혼합(Shrinkage)하여, '0%보다는 높지만 평균보다는 낮은' 합리적인 추정치를 제공합니다.

5.2 Bayesian Logistic Regression (pymc)

`scikit-learn`이 아닌 베이지안 프레임워크 `pymc`를 사용하면 모델을 어떻게 구축할까요? Skittles 맛 테스트 예제를 통해 MCMC가 어떻게 작동하는지 살펴봅니다.

```
import pymc as pm
import numpy as np
import arviz as az

# --- 1. 데이터가정 () ---
# 가지8 다른맛 (x)에 대해, 명n 중명이 y 맛있다고"" 응답
flavoring = np.array([1.69, 1.72, 1.75, 1.78, 1.81, 1.83, 1.86, 1.88])
n_testers = np.array([59, 60, 62, 56, 63, 59, 62, 60])
n_loved = np.array([6, 13, 18, 28, 52, 52, 61, 60])

# --- 2. 모델정의 (with pm.Model() as ...) ---
with pm.Model() as skittles_model:

    # --- 2.1. 사전확률 (Priors) 정의 ---
    # 와alpha 가beta 무엇인지모른다는약한믿음매우 ( 넓은정규분포 )
    alpha = pm.Normal("alpha", mu=0, sigma=100)
    beta = pm.Normal("beta", mu=0, sigma=100)

    # --- 2.2. 모델로직 (Deterministic Link) ---
    # 로지스틱회귀의선형부분로그 ( 오즈)
    logit_p = alpha + beta * flavoring

    # 시그모이드함수를통과시켜확률로 p 변환
    p = pm.Deterministic("p", pm.math.invlogit(logit_p))
```

```
25
26      # --- 2.3. 우도 (Likelihood) 정의 ---
27      # 우리의관측값 (y)이 모델(p)로부터 어떻게생성되었는가 ?
28      # 명n 중의 p 확률로명이 y 성공 -> 이항분포 !
29      y_obs = pm.Binomial("y_obs", n=n_testers, p=p, observed=n_loved)
30
31
32  # --- 3. MCMC 샘플링실행  ---
33  # Metropolis-Hastings 또는( 더발전된  NUTS) 알고리즘이
34  # 복잡한사후분포에서샘플을추출합니다    .
35  # tune: 예열(burn-in) 단계 / draws: 실제저장할샘플수
36  with skittles_model:
37      trace = pm.sample(draws=2000, tune=2000, return_inferencedata=True)
38
39  # --- 4. 결과분석  ---
40  # az.plot_trace(trace, var_names=["alpha", "beta"])
41  # az.summary(trace, var_names=["alpha", "beta"])
```

Listing 2: pymc를 사용한 베이지안 로지스틱 회귀 (Skittles 예시)

5.3 MCMC 결과 해석 (Trace Plot)

MCMC 샘플링 후, `az.summary(trace)`는 다음과 같은 요약 통계량을 보여줍니다.

- **mean, sd:** 추정된 사후분포의 평균과 표준편차. (즉, α는 약 -60.4, β는 약 34.1)
- **hdi_3%, hdi_97%:** 94% 신뢰구간(Credible Interval). 파라미터가 이 범위 안에 있을 확률이 94%라는 의미입니다.
- **ess_bulk, ess_tail:** 유효 샘플 크기. 2000번 뽑았지만, 샘플 간 상관관계 때문에 실제로는 약 1000개 정도의 독립적인 정보만 얻었다는 의미. (높을수록 좋음)
- **r_hat ($\hat{R}$): 가장 중요한 진단 지표.** 여러 개의 MCMC 체인(무작위 탐색가)들이 모두 같은 결과(분포)로 수렴했는지를 봅니다. **정확히 1.0**이거나 1.01 미만이어야만 결과를 신뢰할 수 있습니다. 1.1 이상이면 샘플링이 실패했다는 뜻입니다.

6 학습 체크리스트

- 분류 문제에 왜 선형 회귀 대신 로지스틱 회귀를 사용해야 하는지 설명할 수 있는가?
- 로지스틱 회귀의 계수(β)가 확률이 아닌 '로그 오즈'에 대한 것임을 이해하는가?
- `scikit-learn`에서 `C` 파라미터가 작을수록 정규화가 강해진다는 것을 아는가?
- 데이터가 불균형할 때 왜 정확도(Accuracy) 대신 ROC-AUC를 사용해야 하는지 아는가?
- `np.meshgrid`와 `plt.contourf`를 사용해 결정 경계를 그리는 원리를 이해하는가?
- 베이지안 추론의 3요소 (Prior, Likelihood, Posterior)의 관계를 설명할 수 있는가?
- 켤레 사전분포(예: 베타-이항)가 왜 편리한지 설명할 수 있는가?
- 계층 모델이 '부분적 정보공유(Partial Pooling)'를 통해 과적합을 방지하는 원리(Shrinkage)를 이해하는가?
- 사후분포를 수식으로 풀 수 없을 때 MCMC 샘플링을 사용하는 이유를 아는가?
- MCMC 결과에서 $\hat{R}$ (r_hat) 지표가 1.0에 가까워야 하는 이유를 아는가?

7 FAQ (자주 묻는 질문)

Q: 왜 KNN이나 로지스틱 회귀 전에 `StandardScaler`를 써야 하나요?

A: 이 모델들은 '거리'나 '크기'에 민감하기 때문입니다.
KNN (K-Nearest Neighbors): "가까운 이웃"을 찾을 때 유클리드 거리를 사용합니다. 만약 A 변수(키, 150~190cm)가 B 변수(시험 성적, 0~100점)보다 스케일이 크다면, A 변수가 B 변수보다 거리에 훨씬 큰 영향을 미치게 됩니다. 이는 모델이 사실상 A 변수(키)에만 의존하게 만듭니다.
로지스틱 회귀 (정규화 사용 시): L1/L2 정규화는 계수(β)의 '크기'에 벌칙을 줍니다. 스케일이 큰 변수(A)는 작은 계수($\beta_A \approx 0.01$)를, 스케일이 작은 변수(B)는 큰 계수($\beta_B \approx 10$)를 가질 수 있습니다. 정규화는 β_B에 훨씬 큰 벌칙을 주게 되어 불공평합니다.
`StandardScaler`는 모든 변수의 평균을 0, 표준편차를 1로 만들어, 모든 변수가 공평하게 모델에 기여하도록 만듭니다.

Q: 로그 오즈, 오즈, 확률... 너무 헷갈립니다.

A: 변환의 목적은 '범위'를 맞추는 것입니다.

- **선형 회귀의 출력 (z):** $-\infty$ 에서 $+\infty$ 까지 모든 값을 가집니다.
- **확률 (p):** 0 에서 1 사이의 값만 가져야 합니다.

이 둘을 연결하기 위해, 확률 p를 z와 같은 범위로 바꾸는 변환이 필요합니다.

1. **확률 $\rightarrow$ 오즈:** $Odds = p/(1-p)$. 범위가 $[0,1]$에서 $[0,\infty]$로 바뀝니다. (0보다 작은 값이 없어짐)
2. **오즈 $\rightarrow$ 로그 오즈:** $LogOdds = \ln(Odds)$. 범위가 $[0,\infty]$에서 $[-\infty,+\infty]$로 바뀝니다.

이제 $LogOdds$는 선형 회귀의 출력 z와 범위가 같아졌습니다!

$$\underbrace{\ln(\frac{p}{1-p})}_{\text{범위: } (-\infty,\infty)} = \underbrace{\beta_0 + \beta_1 X}_{\text{범위: } (-\infty,\infty)}$$

로지스틱 회귀는 확률 p가 아닌, **로그 오즈**를 선형적으로 예측하는 모델입니다.

Q: 베이지안 모델은 복잡한데 왜 굳이 사용하나요?

A: 베이지안 모델은 '불확실성의 정량화'와 '계층 구조'라는 강력한 무기를 제공합니다.

- **불확실성 정량화:** 최대우도법(MLE)은 "$\beta_1 = 5.0$"이라는 '점 추정'을 줍니다. 베이지안 모델은 "β_1은 평균 5.0, 표준편차 0.3인 정규분포를 따른다"처럼 '분포 추정'을 줍니다. 즉, β_1이 4.5에서 5.5 사이에 있을 확률이 95%라고 말할 수 있습니다. 이는 우리의 추정이 얼마나 확실한지 알려줍니다.
- **계층 모델 (Shrinkage):** 위에서 설명한 NBA 선수 예시처럼, 데이터가 적은 그룹(선수)이 과적합되는 것을 막아주는 매우 강력하고 합리적인 방법입니다. 이는 일반적인 OLS나 MLE로는 구현하기 매우 어렵습니다.

8 빠르게 훑어보기 (핵심 요약)

로지스틱 회귀 (Logistic Regression)

- **용도:** 선형 회귀($y = ax + b$)의 출력을 시그모이드 함수 $\frac{1}{1+e^{-z}}$에 넣어 0~1 사이의 확률로 변환, '분류' 문제에 사용.
- **핵심:** X가 변할 때 '확률'이 선형적으로 변하는 게 아니라, '로그 오즈' $\ln(p/(1-p))$가 선형적으로 변한다.
- **해석:** β_1 계수는 X가 1 증가할 때 '오즈'가 e^{β_1} **배** 변한다는 의미.
- **손실 함수:** 교차 엔트로피 (Cross-Entropy Loss)

정규화 (Regularization)

- **용도:** 모델이 훈련 데이터에만 과적합(Overfitting)되는 것을 방지.
- **방법:** 계수(β)의 크기에 벌칙(Penalty)을 부과.
- **L1 (Lasso):** $|\beta|$에 비례. 중요하지 않은 변수의 β를 0으로 만듦 (변수 선택).
- **L2 (Ridge):** β^2에 비례. 모든 β를 0에 가깝게 줄임 (부드러운 모델).
- **주의:** `sklearn`의 C는 $1/\lambda$. C가 작을수록 규제가 **강하다**.

베이지안 추론 (Bayesian Inference)

- **핵심 공식:** 사후확률(Posterior) $\propto$ 우도(Likelihood) $\times$ 사전확률(Prior)
- **의미:** 나의 기존 믿음(Prior)을 데이터(Likelihood)를 통해 업데이트(Posterior)한다.
- **켤레성 (Conjugacy):** $Beta(\text{Prior}) + Binomial(\text{Data}) = Beta(\text{Posterior})$ 처럼, 계산이 깔끔하게 떨어지는 조합.
- **사후 평균:** $\hat{p}_{PM} = \frac{a_0+\Sigma y_i}{a_0+b_0+n}$. 사전 믿음과 데이터의 가중 평균.

계층 모델 (Hierarchical Model)

- **용도:** 데이터가 그룹(예: 학생-학급, 선수-팀)으로 묶여 있을 때 사용.
- **개념:** '완전 독립'(No Pooling)과 '완전 동일'(Complete Pooling) 사이의 합리적 절충안.
- **핵심 (Partial Pooling):** 각 그룹(j)의 파라미터(α_j)가 공통 분포 $N(\alpha_0, \sigma_\alpha^2)$에서 나왔다고 가정.
- **효과 (Shrinkage):** 데이터가 적은 그룹(Mac McClung)의 추정치를 전체 평균(α_0) 쪽으로 수축(Shrinkage)시켜 과적합을 방지.

MCMC (Markov Chain Monte Carlo)

- **용도:** 계층 모델처럼 복잡해서 수식으로 풀 수 없는 사후확률(Posterior) 분포를 '샘플링'을 통해 근사적으로 알아내는 방법.
- **Metropolis-Hastings:** '무작위 산책' 알고리즘.
- **로직:** (1) 다음 위치 제안 $\rightarrow$ (2) 현재보다 높으면(Uphill) 무조건 이동 $\rightarrow$ (3) 현재보다 낮으면(Downhill) 확률적으로 이동.

- **결과 해석:** 수렴 진단 지표 $\hat{R}$ (r_hat)이 반드시 1.0에 가까워야 함.