Lecture 11: Frequentist Regression and Bootstrap

Jacob Steinhardt

September 30, 2021
Announcements

- Midterm next Tuesday!
- Review session this weekend (time TBD)
- Probably remote, with recording uploaded
Pitfalls of Bayes

Peril of Bayesian thinking: at the mercy of your model

Poisson distribution too narrow, leads to overconfident posterior

Common issue (esp. with count data): overdispersion
Peril of Bayesian thinking: at the mercy of your model

Poisson distribution too narrow, leads to overconfident posterior

Common issue (esp. with count data): **overdispersion**

Typical fix: negative binomial distribution

\[ p_{\mu, \alpha}(k) \propto \binom{k + \alpha - 1}{k} \left( \frac{\mu}{\mu + \alpha} \right)^k \]

Mean \( \mu \), overdispersion \( \alpha \) (variance \( \mu \cdot \left( 1 + \frac{\mu}{\alpha} \right) \))
Negative binomial plots

[Credit: PyMC3 docs]
Negative binomial regression on turbine data

[Jupyter demo]
Logistic regression revisited

Recall loss function for logistic regression: $L(\theta) = \frac{1}{n} \sum_{i=1}^{n} \ell(x_i, y_i; \beta)$, where

$$\ell(x, y; \beta) = -y \log \sigma(\beta^\top x) - (1 - y) \log(1 - \sigma(\beta^\top x))$$
Logistic regression revisited

Recall loss function for logistic regression: \( L(\theta) = \frac{1}{n} \sum_{i=1}^{n} \ell(x_i, y_i; \beta) \), where

\[
\ell(x, y; \beta) = -y \log \sigma(\beta^\top x) - (1 - y) \log(1 - \sigma(\beta^\top x))
\]

Negative log-likelihood of Bernoulli (coin flip) model:

\[
y \mid x, \beta \sim \text{Bernoulli}(\frac{\exp(\beta^\top x)}{\sum_{j'} \exp(\beta^\top x_{j'})})
\]
Logistic regression revisited

Recall loss function for logistic regression: 

$$L(\theta) = \frac{1}{n} \sum_{i=1}^{n} \ell(x_i, y_i; \beta),$$

where

$$\ell(x, y; \beta) = -y \log \sigma(\beta^\top x) - (1 - y) \log(1 - \sigma(\beta^\top x))$$

Negative log-likelihood of Bernoulli (coin flip) model:

$$y | x, \beta \sim \text{Bernoulli}(\sigma(\beta^\top x))$$
Recall loss function for logistic regression: $L(\theta) = \frac{1}{n} \sum_{i=1}^{n} \ell(x_i, y_i; \beta)$, where

$$\ell(x, y; \beta) = -y \log \sigma(\beta^\top x) - (1 - y) \log(1 - \sigma(\beta^\top x))$$

Negative log-likelihood of Bernoulli (coin flip) model:

$$y \mid x, \beta \sim \text{Bernoulli}(\sigma(\beta^\top x))$$

Logistic regression $\leftrightarrow$ Bernoulli model with sigmoid link function
Logistic regression revisited

Recall loss function for logistic regression: $L(\theta) = \frac{1}{n} \sum_{i=1}^{n} \ell(x_i, y_i; \beta)$, where

$$
\ell(x, y; \beta) = -y \log \sigma(\beta^\top x) - (1 - y) \log(1 - \sigma(\beta^\top x))
$$

Negative log-likelihood of Bernoulli (coin flip) model:

$$
y \mid x, \beta \sim \text{Bernoulli}(\sigma(\beta^\top x))
$$

Logistic regression $\leftrightarrow$ Bernoulli model with sigmoid link function

Why sigmoid? ($\sigma(z) = \frac{1}{1+\exp(-z)} = \frac{\exp(z)}{1+\exp(z)}$)
Logistic regression revisited

Recall loss function for logistic regression: \( L(\theta) = \frac{1}{n} \sum_{i=1}^{n} \ell(x_i, y_i; \beta) \), where

\[
\ell(x, y; \beta) = -y \log \sigma(\beta^\top x) - (1 - y) \log(1 - \sigma(\beta^\top x))
\]

Negative log-likelihood of Bernoulli (coin flip) model:

\[
y \mid x, \beta \sim \text{Bernoulli}(\sigma(\beta^\top x))
\]

Logistic regression \(\leftrightarrow\) Bernoulli model with sigmoid link function

Why sigmoid? \((\sigma(z) = \frac{1}{1 + \exp(-z)} = \frac{\exp(z)}{1 + \exp(z)})\)

- Exponentiate to make positive, normalize to add up to 1
- Generalization: softmax \(\exp(z_j)/\sum_{j'} \exp(z_{j'})\)
(Inverse) Link function + likelihood. Many libraries handle them!

<table>
<thead>
<tr>
<th>Regression</th>
<th>Inverse link function</th>
<th>Link function</th>
<th>Likelihood</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear</td>
<td>identity</td>
<td>identity</td>
<td>Gaussian</td>
</tr>
<tr>
<td>Logistic</td>
<td>sigmoid</td>
<td>logit</td>
<td>Bernoulli</td>
</tr>
<tr>
<td>Poisson</td>
<td>exponential</td>
<td>log</td>
<td>Poisson</td>
</tr>
<tr>
<td>Negative binomial</td>
<td>exponential</td>
<td>log</td>
<td>Negative binomial</td>
</tr>
</tbody>
</table>
Recap

- Bayesian regression
  - Least squares = MLE
  - Ridge regression = MAP

- Overdispersion
  - Model mis-specification \(\implies\) overly narrow uncertainty

Next up: model checking and frequentist GLMs
Recap

- Bayesian regression
  - Least squares = MLE
  - Ridge regression = MAP
- Overdispersion
  - Model mis-specification $\Rightarrow$ overly narrow uncertainty

Next up: model checking and frequentist GLMs
Posterior Predictive Distribution

Have so far seen several Bayesian objects:

- Prior $p(\theta)$
- Posterior $p(\theta | x_{1:n})$

Another important object: **posterior predictive distribution** (PPD)

- Predict a new data point from data so far
- E.g.: $p(x_{n+1} | x_{1:n})$, or $p(y_{n+1} | x_{n+1}, y_{1:n}, x_{1:n})$
Trick for computing PPD

Suppose that $x_{n+1} \perp x_1, \ldots, x_{n+1} \mid \theta$ (Bayesian hierarchical model)

Then:

$$p(x_{n+1} \mid x_{1:n}) = \int \int p(x_{n+1} \mid \theta) p(\theta \mid x_{1:n}) d\theta$$

(1)

⇒ Draw samples $\theta$ from posterior, then sample $x_{n+1}$ conditioned on $\theta$.

(or just average $p(x_{n+1} \mid \theta)$ to get density)
Trick for computing PPD

Suppose that $x_{n+1} \perp \perp x_1, \ldots, x_{n+1} \mid \theta$ (Bayesian hierarchical model)

Then:

$$p(x_{n+1} \mid x_{1:n}) = \int \int p(x_{n+1} \mid \theta) \underbrace{p(\theta \mid x_{1:n})}_{	ext{posterior for } \theta} \, d\theta$$  \hspace{1cm} (1)

$\implies$ Draw samples $\theta$ from posterior, then sample $x_{n+1}$ conditioned on $\theta$.

(or just average $p(x_{n+1} \mid \theta)$ to get density)

More general idea: add $x_{n+1}$ as a variable in PyMC3, and just sample it along with everything else

- Pro: works for any model structure
- Con: have to decide in advance which predictions you care about
PPD gives us ways of **checking** a model.

Intuition: check that predicted data “looks like” real data

Examples:
- Check that predicted $y_i$ look like true $y_i$ in regression
- Check predictions on a hold-out set
Brainstorming exercise: What are ways we could formalize whether predicted samples “look like” real data?

(What statistics would you compute?)
[Demo with wind turbine data]
Frequentist GLMs

So far, looked at GLMs in Bayesian framework: prior + posterior + inference

Works equally well in frequentist world: just drop prior and use MLE!

\[ \hat{\beta}_{\text{MLE}} = \arg\min_{\beta} \frac{1}{n} \sum_{i=1}^{n} \log \text{Poisson}(y_i; \exp(\beta^\top x_i)) \]
Frequentist GLMs

So far, looked at GLMs in Bayesian framework: prior + posterior + inference

Works equally well in frequentist world: just drop prior and use MLE!

I.e. for Poisson:

\[
\hat{\beta}_{\text{MLE}} = \arg\min_{\beta} \frac{1}{n} \sum_{i=1}^{n} \log P_{\text{Poisson}}(y_i; \exp(\beta^T x_i))
\]  

(2)
Frequentist GLMs

So far, looked at GLMs in Bayesian framework: prior + posterior + inference

Works equally well in frequentist world: just drop prior and use MLE!

I.e. for Poisson:

$$\hat{\beta}_{\text{MLE}} = \arg\min_{\beta} \frac{1}{n} \sum_{i=1}^{n} \log P_{\text{Poisson}}(y_i; \exp(\beta^T x_i))$$  \hspace{1cm} (2)

There is also a package for handling this: statsmodels.
[Jupyter wind turbine demo]
Model Checking for MLE

No prior, so no posterior. Are there other types of predictive checks we can use?

[Brainstorming exercise]
No prior, so no posterior. Are there other types of predictive checks we can use?

[Brainstorming exercise]

Example: Chi-square statistic

\[
\sum_{i=1}^{n} \frac{(y_i - \mathbb{E}[y \mid x_i, \hat{\beta}])^2}{\text{Var}[y \mid x_i, \hat{\beta}]} \tag{3}
\]
Frequentist Model Checking: Summary

- Log-likelihood: larger (less negative) for better models

- Deviance: for $n$ datapoints and $p$ parameters, should be roughly $n - p$ (assuming model is correct)

- Chi-square statistic: also should be roughly $n - p$. (Why not $n$?)
Many tools for model checking:

- Bayesian: posterior predictive checks
- Frequentist: chi-square statistic

All models are “wrong” to some degree. These tools tell us if things are “obviously” wrong.