Machine Learning and Data Mining - Homework 1

Philip Warton

October 13, 2021

1 Statistical Estimation

1.1 Q1

Let $D = \{x_1, \cdots, x_N\}$ be a dataset from N poisson random variables, with a rate of $\lambda \in \mathbb{R}$. Derive the Maximum Likelihood Estimation for λ.

We begin by taking $\mathcal{L}(D)$. That is,

$$\begin{aligned}
\mathcal{L}(D) &= P(D \mid \lambda) \\
&= P(\{x_1, \cdots, x_N\} \mid \lambda) \\
&= P(x_1 \mid \lambda) \cdots P(x_N \mid \lambda) \\
&= \prod_{i=1}^{N} P(x_i \mid \lambda) \\
&= \prod_{i=1}^{N} \frac{\lambda^{x_i} e^{-\lambda}}{x_i!}
\end{aligned}$$

Then to get the log-likelihood, we take $\ln \mathcal{L}(D)$.

$$\begin{aligned}
\ln \mathcal{L}(D) &= \ln \prod_{i=1}^{N} \frac{\lambda^{x_i} e^{-\lambda}}{x_i!} \\
&= \sum_{i=1}^{N} \ln \left(\frac{\lambda^{x_i} e^{-\lambda}}{x_i!} \right) \\
&= \sum_{i=1}^{N} \left[\ln(\lambda^{x_i} e^{-\lambda}) - \ln(x_i!) \right] \\
&= \sum_{i=1}^{N} \left[\ln(\lambda^{x_i}) + \ln(e^{\lambda}) - \sum_{j=1}^{i} \ln(x_j) \right] \\
&= \sum_{i=1}^{N} \left[x_i \ln \lambda - \lambda \ln e - \sum_{j=1}^{i} \ln x_j \right] \\
&= \ln \lambda \sum_{i=1}^{N} x_i - N\lambda - \sum_{i=1}^{N} \sum_{j=1}^{i} \ln x_j
\end{aligned}$$

We now take the derivative of the log-likelihood, giving us

$$\begin{aligned}
\frac{d}{d\lambda} \left(\ln \mathcal{L}(D) \right) &= \frac{d}{d\lambda} \left(\ln \lambda \sum_{i=1}^{N} x_i - N\lambda - \sum_{i=1}^{N} \sum_{j=1}^{i} \ln x_j \right) \\
&= \frac{1}{\lambda} \sum_{i=1}^{N} x_i - N - 0
\end{aligned}$$

Let this derivative be equal to zero. Then we have

$$
\begin{aligned}
0 &= \frac{1}{\lambda}\sum_{i=1}^{N} x_i - N \\
N &= \frac{1}{\lambda}\sum_{i=1}^{N} x_i \\
\lambda &= \frac{1}{N}\sum_{i=1}^{N} x_i \\
\lambda &= \overline{x}
\end{aligned}
$$

1.2 Q2

The log posterior is

$$\log P(\lambda \mid D) \propto P(D \mid \lambda) + P(\lambda) = \prod_{i=1}^{N} \frac{\lambda^{x_i} e^{-\lambda}}{x_i!} + \frac{\beta^{\alpha}\lambda^{\alpha-1}e^{-\beta\lambda}}{\Gamma(\alpha)}$$

We take the derivative of the log posterior.

$$
\begin{aligned}
(d/d\lambda)\log P(\lambda \mid D) &= (d/d\lambda)c\log P(D \mid \lambda) + \log(\text{Gamma of lambda}) \\
&= \frac{1}{\lambda}\sum_{i=1}^{N} x_i - N + \frac{d}{d\lambda}\left(\ln\left(\frac{\beta^{\alpha}\lambda^{\alpha-1}e^{-\beta\lambda}}{\Gamma(\alpha)}\right)\right) \\
&= \frac{1}{\lambda}\sum_{i=1}^{N} x_i - N + \frac{d}{d\lambda}\Big(\ln(\beta^{\alpha}) + \ln(\lambda^{\alpha-1}) + \ln(e^{-\beta\lambda}) - \ln(\Gamma(\alpha))\Big) \\
&= \frac{1}{\lambda}\sum_{i=1}^{N} x_i - N + \frac{d}{d\lambda}\Big(\alpha\ln(\beta) + (\alpha-1)\ln(\lambda) + -\beta\lambda - \ln(\Gamma(\alpha))\Big) \\
&= \frac{1}{\lambda}\sum_{i=1}^{N} x_i - N + \left(0 + (\alpha-1)\frac{1}{\lambda} + -\beta - 0\right) \\
&= \frac{1}{\lambda}\sum_{i=1}^{N} x_i - N + \frac{\alpha-1}{\lambda} + -\beta \\
&= \frac{1}{\lambda}\left(\left(\sum_{i=1}^{N} x_i\right) + \alpha - 1\right) - (N+\beta)
\end{aligned}
$$

Now let this derivative be equal to 0. Then,

$$
\begin{aligned}
0 &= \frac{1}{\lambda}\left(\left(\sum_{i=1}^{N} x_i\right) + \alpha - 1\right) - (N+\beta) \\
N+\beta &= \frac{1}{\lambda}\left(\left(\sum_{i=1}^{N} x_i\right) + \alpha - 1\right) \\
\lambda &= \frac{1}{N+\beta}\left(\left(\sum_{i=1}^{N} x_i\right) + \alpha - 1\right) \\
\lambda &= \frac{\left(\sum_{i=1}^{N} x_i\right) + \alpha - 1}{N+\beta}
\end{aligned}
$$

And thus we have a closed form solution for our parameter λ.

1.3 Q3

We write

$$\left(\prod i = 1^{N} \frac{\lambda^{x_i} e^{-\lambda}}{x_i!}\right)\left(\frac{\beta^{\alpha}\lambda^{\alpha-1}e^{-\beta\lambda}}{\Gamma(\alpha)}\right) = \left(\frac{\beta}{\Gamma(\alpha)\prod x_i} \cdot \lambda^{(\sum x_i + \alpha)-1} \cdot e^{-(N+\beta)\lambda}\right) \approx \text{Gamma}$$

2 k-Nearest Neighbor Classifier

2.1 Q4

Consider the case where we have one sample x such that workclass$[x]$ = Private and where all other variables are 0. Then we have another sample y such that workclass$[y]$ = Never-worked and all other variables are 0. Within our first encoding, we have

$$||x - y|| = \sqrt{0 + \cdots + 1^2 + 0 + 1^2 + \cdots + 0} = \sqrt{2}$$

Whereas in the second encoding, where we simply keep one column but associate different integers with category we would have

$$||x - y|| = \sqrt{0 + \cdots + (3-1)^2 + \cdots + 0} = 2$$

This discrepancy will grow larger as the number of categories in a given column grows larger, and what we see is certain categorical differences will be a greater difference than other categorical differences within that same column. For this reason, we prefer binarization so that the categories are evenly spaced, even though we may have to work in a higher dimension.

2.2 Q5

We use the following code to compute the percentage of people with an income larger than 50 thousand dollars per year:

```
def printPercentOver50k(D):
    k = 0
    for x in D:
        if (x[85] == 1):
            k += 1
    n = 8000
    print(100 * (k / n))

printPercentOver50k(D)
```

This gives us a result of 24.5875%. We know that 70% accuracy will not be very good because we could achieve this accuracy by classifying each datapoint as 0. We want our model to have a much higher accuracy than this.

2.3 Q6

Proof. Let $x, z \in \mathbb{R}^n$.

$$\begin{aligned} d(x,z) &= \sqrt{(x_1 - z_1)^2 + (x_2 - z_2)^2 + \cdots + (x_n - z_n)^2} \\ &= \sqrt{\sum_{i=1}^{n} (x_i - z_i)^2} \\ &= ||x - z||_2 \end{aligned}$$

□

2.4 Q8

2.5 Q9

The best number of k I observed would be $k = 1$ or $k = 9$. For $k = 1$ it had good accuracy on the training data, but not so much on the k-fold cross validation. As k increased, training accuracy decreased and k-fold cross validation accuracy fluctuated somewhat. For $k = 1$, our training error was minute, but not zero. This is because we have some duplicate points that may have different outputs.

2.6 Q10

3 Status Report

```
~15.5h
Difficult
Discussed the problems with others
I understand the material about 20%
No other comments
```