10/3/2021

CS 434 HW1

Gretel Rajamoney

Q1: the likelihood function of λ given a dataset $D = \{x_1, x_2, \ldots, x_n\}$ is

$$P(D|\lambda) = \prod_{i=1}^{N} e^{-\lambda} \cdot \frac{\lambda^{x_i}}{(x_i)!} = e^{-N\lambda} \cdot \frac{\lambda^{\sum_{i=1}^{N} x_i}}{\prod_{i=1}^{N}(x_i!)}, \quad x = 0,1,2,3,\ldots$$

the log-likelihood function of λ is found by $\log(P(D|\lambda))$ and is:

$$\log_e P(D|\lambda) = \log_e\left[e^{-N\lambda} \cdot \frac{\lambda^{\sum_{i=1}^{N} x_i}}{\prod_{i=1}^{N}(x_i!)}\right]$$

$$= \log_e(e^{-N\lambda}) + \log_e(\lambda^{\sum_{i=1}^{N} x_i}) - \log_e\left(\prod_{i=1}^{N}(x_i!)\right)$$

$$= -N\lambda + \sum_{i=1}^{N} x_i(\log_e \lambda) - \sum_{i=1}^{N} \log_e(x_i!)$$

next we take the derivative with respect to parameter λ:

$$\frac{d}{d\lambda}\left[\log_e P(D|\lambda)\right] = \frac{d}{d\lambda}\left[-N\lambda + \sum_{i=1}^{N} x_i(\log_e \lambda) - \sum_{i=1}^{N} \log_e(x_i!)\right]$$

$$= -N + \frac{1}{\lambda}\sum_{i=1}^{N} x_i$$

next we solve for the maximizing value $\hat{\lambda}_{MLE}$, $\frac{d}{d\lambda}\left[\log_e(P(D|\lambda))\right] = 0$:

$$-N + \frac{1}{\lambda}\sum_{i=1}^{N} x_i = 0$$

$$= \frac{1}{\lambda}\left(-N\lambda + \sum_{i=1}^{N} x_i\right) = 0$$

$$= -N\lambda = -\sum_{i=1}^{N} x_i$$

$$\lambda = \frac{1}{N}\sum_{i=1}^{N} x_i$$

$$\lambda = \bar{x}$$

therefore, $\hat{\lambda}_{MLE} = \bar{x}$

since in poisson distribution λ is the average number of occurrences in an interval, we can intuitively assume that λ can be estimated by the mean number of occurrences within the sample data

Q2: the log posterior of λ is $\log(P(\lambda|D)) \propto \log(P(D|\lambda)) + \log(P(\lambda))$

$$f(x) = \frac{e^{-\lambda\beta} \cdot \lambda^{\lambda-1}}{(\alpha)} \cdot \beta^{\alpha}, \quad \lambda > 0$$

$$f(x_i, \lambda) = \prod_{i=1}^{N} f(x_i|\lambda) f(x)$$

$$= e^{-N\lambda} \cdot \frac{\lambda^{\sum_{i=1}^{N} x_i}}{\prod_{i=1}^{N} \lfloor x_i} = \frac{e^{-\lambda\beta} \lambda^{\alpha-1}}{p(\alpha)} \cdot \beta^{\alpha}$$

$$= \int_0^{\infty} \frac{\beta^{\alpha}}{\prod_{i=1}^{N} \lfloor x_i | \alpha} = e^{-N\lambda} \cdot e^{-\lambda\beta} \cdot \lambda^{\sum_{i=1}^{N} x_i} \cdot \lambda^{\alpha-1} \frac{d}{d\lambda}$$

$$= \frac{\beta^{\alpha}}{\prod_{i=1}^{N} \lfloor x_i \sqrt{\alpha}} \cdot \int_0^{\infty} -\lambda(N+\beta) \cdot \lambda^{\sum_{i=1}^{N} x_i + \alpha - 1} \frac{d}{d\lambda}$$

$$p(x|0) = \frac{f(x) : \lambda}{f(x)}$$

next we must take the derivative

$$\frac{d}{d\lambda} \cdot P(\lambda|D) = 0 + \frac{d}{d\lambda}\left[-\lambda(N+\beta)\right] + \frac{d}{d\lambda}\left\{h = \lambda\left[\sum_{i=1}^{N} x_i + \alpha - 1\right]\right\}$$

$$= -(N+\beta) + \frac{\sum_{i=1}^{N} x_i + \alpha - 1}{\lambda}$$

now we set the derivative to zero and solve for $\hat{\lambda}_{MAP}$

$$-(N+\beta) + \frac{\sum_{i=1}^{N} x_i + \alpha - 1}{\lambda} = 0$$

$$= \sum_{i=1}^{N} x_i + \alpha - 1 = \lambda(N+\beta)$$

$$= \frac{\sum_{i=1}^{N} x_i + \alpha - 1}{N+\beta}$$

$$\lambda = \hat{\lambda}$$

the maximum a posteriori estimator of λ is $\frac{\sum_{i=1}^{N} x_i + \alpha - 1}{N+\beta}$

Q3: the conditional joint density of $y_1, y_2, \ldots, y_N | \lambda$ is:

$$f(y_1, y_2, \ldots, y_N|\lambda) = \prod_{i=1}^{N} \left(\frac{e^{-\lambda} \cdot \lambda^{y_i}}{y_i!} \right) = \frac{e^{-N\lambda} \cdot \lambda^{\sum_{i=1}^{N} y_i}}{\prod_{i=1}^{N} y_i!}$$

$$\pi(\lambda) = \frac{e^{-\beta\lambda} \cdot \lambda^{\alpha-1} \cdot \beta^{\alpha}}{\Gamma\alpha}, \; \lambda > 0$$

therefore the posterior distribution is:

$$\pi(\lambda|D) = \pi(\lambda|y_1, y_2, \ldots y_N) = \frac{f(y_1, y_2, \ldots y_N|\lambda)\,\pi(\lambda)}{\int_0^\infty f(y_1, y_2, \ldots, y_N|\lambda)\,\pi(\lambda)\,d\lambda}$$

meaning that $\int_0^\infty f(y_1, y_2, \ldots, y_N)\pi(\lambda)d\lambda$ is independent of λ and therefore is constant

$$= \frac{e^{-N\lambda} \cdot \lambda^{\sum_{i=1}^{N} y_i}}{\left(\prod_{i=1}^{N} y_i!\right)} \cdot \frac{e^{-\beta\lambda} \cdot \lambda^{\alpha-1} \cdot \beta^{\alpha}}{\Gamma\alpha} \propto e^{-\lambda(\beta+N)} \cdot \lambda^{\left(\sum_{i=1}^{N} y_i + \alpha - 1\right)}$$

$\lambda|D \sim \text{gamma}(\alpha_p, \beta_p)$ where $\alpha_p = \sum_{i=1}^{N} y_i + \alpha - 1$ and $\beta_p = \beta + N$

therefore gamma distribution is a conjugate prior to the poisson distribution

Q4:

Ordinal Encoding is when each category within a categorical variable is assigned an integer value. One-Hot Encoding is when each category within a categorical variable is separated and assigned binary values (0 or 1).

Utilizing ordinal encoding would not work (or atleast be incredibly inefficient) to utilize when it comes to Euclidean distance in kNN. When calculating the distance, you need to utilize numerical values that carry accurate weight. In terms of calculating Euclidean distance in kNN, the one-hot encoding approach would be optimal. This is because their numerical values actually carry weight when using them for calculations.

Q5: sum = 1967
count = 8000

total % with income >50k $= \frac{1967}{8000} = 24.59\%$

Approximately 25% of individuals represented in the training data have incomes greater than \$50k. A model that achieves 70% accuracy is considered to be a good model based on online sources. Each data point has 12 dimensions excluding the increased dimensionality that one-hot encodings bring. When including the increased dimensionality that one-hot encodings bring, each data point has 85 dimensions (ignoring the id attribute and class label).

Q6: let $x = (x_1, x_2, \ldots, x_d)^T$ be x vector,
next, zero vector: $x = (0, 0, \ldots, 0)^T$

the euclidean distance between vector x & the zero vector is:

$$\sqrt{(x_1-0)^2+(x_2-0)^2+\ldots+(x_d-0)^2}$$
$$= \sqrt{\sum_{i=1}^{d} x_i^2} = ||x||_2$$

let $z = (z_1, z_2, \ldots, z_d)^T$

the euclidean distance between vector x & vector z is:

$$\sqrt{(x_1-z_1)^2+(x_2-z_2)^2+\ldots+(x_d-z_d)^2}$$
$$= \sqrt{\sum_{i=1}^{d} (x_i-z_i)^2} = ||x-z||_2$$

therefore, the distance between vector x & vector z can be written in the form of L_2 norm

Debriefing

1. I spent approximately 22 hours on this assignment, this was mainly because I am very unexperienced in Python, and this algorithm took a really long time for me to understand thoroughly.

2. I would rate this assignment very difficult. The factors I mentioned above are the main reasoning for my rating. I also struggled slightly understanding how kaggle worked as well as interpretting what Q4 was asking for. I am sure that students more familiar with Python & algorithms would rank this assignment lower difficulty than I did.

3. I worked on this assignment alone, but I did view YouTube videos as well as our textbook chapters heavily in order to understand the kNN algorithm and k-fold cross validation.

4. 90%

5. Not sure if this is an appropriate question to ask, but in comparison to this assignment, how much more difficult will the upcoming assignments be?
(asking solely so that I can allot time accordingly, and ensure that I seek help well in advance)