CAP 6610 Machine Learning

Siddhant Mittal (6061-8545)

1. Given the uniform distribution for a continuous variable x, $p(x; a, b) = \frac{1}{b-a}$ where $a \leq x \leq b$. This distribution is normalized can be checked by integrating it over (a,b) if it's result is 1 that means it is normalized.

$$\int_a^b p(x; a, b) = \int_a^b \frac{1}{b-a} dx$$

$$\frac{b-a}{b-a} = 1$$

Mean and variance for a function defined over continuous variable can be defined as $E[X]$ and $E[X^2] - E[X]^2$ respectively.

Expression for mean:

$$E[X] = \int_a^b \frac{x}{b-a} dx = \frac{b^2 - a^2}{2(b-a)} = \frac{b+a}{2}$$

Expression for variance:

$$E[X^2] - E[X]^2 = \int_a^b \frac{x^2}{b-a} dx - \frac{(b+a)^2}{4}$$

$$\frac{b^3 - a^3}{3(b-a)} - \frac{(b+a)^2}{4}$$

$$\frac{b^2 + ab + a^2}{3} - \frac{(b+a)^2}{4} = \frac{(b-a)^2}{12}$$

2. Given Probability Mass Function(PMF) of a possion distribution

$$p(x; \lambda) = \frac{\lambda^x e^{-\lambda}}{x!}$$

Given Independent and identically distributed sample $x_1, ..., x_n \sim Pois(\lambda)$. Maximum Likelihood Estimation for λ can we calculated as:

Probability of observing $x_1, ..., x_n =$

$$P(x_1; \lambda)P(x_2; \lambda).....P(x_n; \lambda)$$

Since $x_1, ..., x_n \sim Pois(\lambda)$ is i.i.d.

$$L = \frac{\lambda^{\sum_i x_i} e^{-n\lambda}}{x_1!....x_n!}$$

$$l = logL = -n\lambda + (\sum_i x_i)\log\lambda + C$$

where $c = -\log(\frac{1}{x_1!....x_n!})$ that is not dependent on λ.

$$\frac{dl}{d\lambda} = -n + \frac{(\sum_i x_i)}{\lambda} = 0$$

$$\lambda = \frac{(\sum_i x_i)}{n}$$

3. Given randn(d,1) which gives multivariate normal variable $x \in R^d$) which are in N(0,I). We want to generate random variable X = $x_1, ..., x_n$ for N(μ,Σ).
   Let Z = $(Z_1, ..., Z_n)^T$ where Z_i is generated using randn(d,1)

   $$X \sim N(\mu, \Sigma)$$

   $$Z \sim N(\mu, \Sigma); \mu = 0, \Sigma = I$$

   If C is an n*d matrix

   $$C^T Z \sim N(C^T\mu, C^T\Sigma C)$$

   Our problem therefore reduces to finding such C that $C^T C = \Sigma$, we can use the cholesky decomposition of Σ to find such matrix. So algorithm to generate random variable can be divided into following steps.

   (a) Generate Z using randn(d,1).
   (b) Now compute the cholesky decomposition to get C.
   (c) X = $C^T Z$

4. To find an expression for $\overline{\theta}$ that minimise loss function we take derivative with respect to θ (gradient) and equate it to zero.

$$f(x) = \frac{1}{n}\sum_{i}^{n} x_i(y_i - \phi_i^T\theta)^2$$

$$\frac{df(x)}{d\theta} = 0$$

$$-\sum_{i}^{n} x_i(y_i - \phi_i^T\theta)\phi_i = 0$$

$$-\sum_{i}^{n} x_i y_i \phi_i + (\sum_{i}^{n} x_i\phi_i\phi_i^T)\theta = 0$$

$$\theta = (\sum_{i}^{n} x_i\phi_i\phi_i^T)^{-1}(\sum_{i}^{n} x_i y_i \phi_i)$$

5. Classifier A
   Since we are ignoring word count we have like Bernoulli random distribution P of taking x is either 0 and 1. Probability of document d belonging to class j is equal to multiplication of probability of each word belonging to that class. I have taken log of values to avoid overflow.

$$P(D/y_i) = \log(CDP(y_j)) + \log(\prod_{i=1}^{n} P(x_i/y_i))$$

$$\hat{y} = max(P(D/y_1), P(D/y_2)...P(D/y_n))$$

   where,
   CDP - Class Distribution Probability
   List of words as feature vectors
   D = document = $[x_1, ..., x_n]$
   Prediction accuracy on 20 Newsgroups test data set is 79%

   Classifier B
   Since now we have taken into account the frequency of words, we made our feature vector multinomial. So we model the classifier now as multinomial Naive Bayes Classifier.

$$P(D/y_i) = \log(CDP(y_j)) + \log(\prod_{i=1}^{n}(P(x_i/y_i))^{f(D,x_i)})$$

$$\hat{y} = max(P(D/y_1), P(D/y_2)...P(D/y_20))$$

where,
CDP - Class Distribution Probability
P(x_i/y_j) = Probability of x_i belongs to y_j class
f(D, x_i) = word count of x_i in D.

Prediction accuracy on 20 Newsgroups test data set is 75%

Classifier C

Bag of words may contain stop words(common words) and we do a pre-processing in classifier B to make the data discrete. But if we want our Multinomial naive bayes to work like LDA we have to make our feature vector as continuous variable with same co-variance matrix.

So to facilitate this we add Inverse Document Frequency weight on each word.

$$t_i = \log(\sum_{n=1}^{N} doc_n/doc_i)$$

$$P(D/y_i) = \log(CDP(y_i)) + \sum i = 1^n f(d, x_i) \log(t_{xi} P(x_i/y_i))$$

Prediction accuracy on 20 Newsgroups test data set is 75%