CS60050 Machine Learning - Weekly Report

Nisarg Upadhyaya (19CS30031)

Week 7: 22-24 September

1 Topics covered

- Parametric Methods
- Bias, MSE and Variance of Estimators
- The Bayes' Estimator
- Parametric Classification and Multivariate Representation
- Generalisation to Multinomial Cases
- Non-parametric Approaches
- Non-parametric Classification
- Instance Based Learning
- Unsupervised Learning and Clustering
- K-means Clustering

2 Summary

2.1 Parametric Methods

Here we assume that the sample is drawn from some distribution that obeys a known model which can be described by some parameters. Once the parameters are known we can get probability measures and make decisions. Let the dataset be $X = \{x^t\}, t = 1, 2, ...N$ where each sample is independent and identically distributed. Let the parameters be θ then we try to maximise the log-likelihood $L(\theta|X) = \sum_{t=1}^{N} logP(x^t|\theta)$. For a few standard distributions we have:

1. Bernoulli: Two possible outcomes of X, 1 and 0. $P(x) = p^x(1-p)^{(1-x)}$. Single parameter p. Likelihood is maximised when $p = m = \frac{1}{N}\sum_{t=1}^{N} x^t$ where m is also called the sample average.
2. Multinomial: K possible outcomes of X. $P(x) = \prod_{i=1}^{K} p_i^{x_i}$. K parameters $p_1, p_2, ...p_K$. Likelihood is maximised when $p_i = \frac{1}{N}\sum_{t=1}^{N} x_i^t$.
3. Normal: For normal distribution with parameters μ and σ^2 likelihood is maximised when $\mu = \frac{1}{N}\sum_{t=1}^{N} x^t$ and $\sigma^2 = s^2 = \frac{1}{N}\sum_{t=1}^{N}(x^t - \mu)^2$ where s^2 is also called the sample variance.

2.2 Bias, MSE and Variance of Estimators

Let X be a sample and $d(X)$ be an estimator of θ. We have the bias of the estimator $b_\theta(d) = E[d(X)] - \theta$. Consider the sample average m. We have $E[m] = \mu$ and $Var(m) = \frac{\sigma^2}{N}$. So m is an unbiased and also a consistent estimator of mean μ. If we consider the sample variance s^2 then we have $E[s^2] = \frac{N-1}{N}\sigma^2$. This is an asymptotically unbiased estimator. The MSE of an estimator $d(X)$ is given by $r(d, \theta) = E[(d(X) - \theta)^2]$ which can be simplified to $r(d, \theta) = Var(d) + (b_\theta(d))^2$.

2.3 The Bayes' Estimator

The prior density $p(\theta)$ tells us the likely values θ takes before looking at the sample. Combining this with the likelihood density $p(X|\theta)$ we get the posterior density $p(\theta|X)$ which tells us the likely values of θ after looking at the sample. We have:

$$p(\theta|X) = \frac{p(X|\theta)p(\theta)}{\int p(X|\theta')p(\theta')d\theta'}$$

Another estimation is the *Bayes' Estimator* which is defined as the expected value of posterior density, $\theta_{Bayes} = \int \theta p(\theta|X)d\theta$.

2.4 Parametric Classification and Multivariate Representation

From the Bayes' Classifier we know that we assign a class C_i to an instance x which has the maximum posterior $P(C_i|x)$ with the discriminant function $g_i(x) = logP(x|C_i) + logP(C_i)$. If we assume $P(x|C_i)$ is Gaussian then we can write the discriminant function as:

$$g_i(x) = -\frac{1}{2}log(2\pi) - log(\sigma_i) - \frac{(x-\mu_i)^2}{2\sigma_i^2} + log(P(C_i))$$

For the multivariate case where each $x^t \in R^d$ we construct a data matrix X where each row is a data sample. The mean vector μ is the mean of each column and the covariance matrix $\Sigma = E(X^T X) - \mu^T \mu$. In this case we can write the discriminant function as:

$$g_i(x) = -\frac{d}{2}log(2\pi) - \frac{1}{2}log(|\Sigma_i|) - \frac{1}{2}(x-\mu)^T \Sigma_i^{-1}(x-\mu) + log(P(C_i))$$

2.4.1 Generalisation to Multinomial Cases

If each x_j can take one of n_j distinct values $v_1, v_2, ...v_{n_j}$ then we define a dummy variable z_{jk} which is 1 only if $x_j = v_k$ otherwise 0. Then we have the parameter $p_{ijk} = P(z_{jk} = 1|C_i)$. In this case we have the discriminant function as $\sum_j \sum_k z_{jk} log(p_{ijk}) + log(P(C_i))$.

2.5 Non-parametric Approaches and Classification

In this we assume similar inputs have similar outputs and follow an instance based or memory based learning. For univariate nonparametric density estimation one approach is the *histogram estimator* where the input space is divided into equal-sized bins. For a bin width h we have $p(x) = \frac{\#\{x^t \text{ in the same bin as } x\}}{Nh}$. For a kernel estimator, also called *Parzen Windows* we have $p(x) = \frac{1}{Nh}\sum_{t=1}^{N} K(\frac{x-x^t}{h})$ where $K(\cdot)$ is the kernel function. We can also use the *k-nearest neighbor* density estimate as $p(x) = \frac{k}{2Nd_k(x)}$ where $d_k(x)$ is the distance to the k-th nearest sample. We can use an adaptive kernel estimator which is the kernel estimator with $h = 2d_k(x)$. For non parametric classification we have the discriminant function as $g_i(x) = \frac{1}{Nh^d}\sum_{t=1}^{N} K(\frac{x-x^t}{h}) r_i^t$ where r_i^t is 1 if $x^t \in C_i$ otherwise 0.

2.6 Instance Based Learning

In this we store the set of instances while training and for queries we retrieve similar set of related instances and classify/regress using those. In *k-NN Regression* for training examples $(x^t, f(x^t))$ we regress by taking the k-nearest neighbors of x, i.e., $\hat{f}(x) = \frac{\sum_{i=1}^{k} f(x_i)}{k}$. In locally weighted regression we take target function to be linear on attribute values $f(x) = w_0 + w_1 x_1 + w_2 x_2 + ... w_d x_d$ where the weights are chosen that minimise the squared error sum over the training examples.

2.7 Unsupervised Learning and Clustering

In this no labels are provided. We learn only from the input data and the aim is to find similarities/structure in the given input. Clustering is the task of organising *similar* (in some way) objects into groups. Note that a cluster is a group with *loosely* defined similarity among the objects unlike a class which is well defined.

2.8 K-means Clustering

In this we are given N d-dimensional datapoints. We want to compute k-partitions, where each partition is represented by its center and minimise the sum of square of distances between a data point and the center of its respective cluster. Enumerating all possible ways to distribute N objects into k distinct groups will take exponential time. Instead we use **The Lloyd algorithm**. Given k initial centers it assigns a data point the cluster center which is closest to it. It then updates the centers based on this assignment. These steps are repeated till the centers do not change their position. A more conservative approach would be to move data point at a time provided overall cost gets reduced. While this algorithm is simple and tries to minimise the sum of divergences of each cluster from its center it is highly sensitive to the selection of initial points and noise which may result in empty clusters and bad local minimas. There are various initialisation approaches like assigning each point randomly to one of the clusters, taking first k points as centers, k-means on random subsets, etc. There is also **k-means++** which chooses first center randomly and the i-th one as x' with a probability proportional to square of the minimum distance from the selected $i-1$ centers.

3 Challenging concepts

Parametric classification and generalisation to multinomial cases.

4 Interesting concepts

Parametric estimation and k-means clustering.

5 Concepts not understood

None

6 Ideas

In general we don't have prior information in unsupervised learning. However, say we have some information available associated with the data we are trying to classify, e.g., several news articles are present and we know what topic each one is on, then if we apply k-means on such a dataset to try to classify articles into different categories, once done we can also by some heuristic, say majority vote, decide what a cluster center represents or the percentage probability of finding an article of a particular type in that cluster by checking the articles which belong to that cluster. This way when a new instance is assigned a cluster we can now assign a confidence score to it being a particular type of article.