Non Parametric Project (MTH-516A) Report

Naive Bayes Using Kernel Density Estimation

Submitted by

Roll No	Names of Students
201261	Akash Kumar Sharma
201279	Arkonil Dhar
201433	Souvik Bhattacharyya

Under the guidance of
Dr. Dootika Vats

Department of Mathematics and Statistics
INDIAN INSTITUTE OF TECHNOLOGY KANPUR

ABSTRACT

In classical nave bayes approach we assume that the data are generated by a single Guassian. In this project we abandon the normality assumption and instead use statistical methods for non-parametric density estimation.In this project, we have applied nave bayes approach on both real world and simulated data comparing two methods of density estimation: assuming normality and modeling each conditional probability distribution with a single Gaussian; and using non-parametric kernel density estimation. We observe that when the data sets are not from the Gaussian distribution there are large reductions in error on several real world and simulated data sets, which suggests that kernel estimation is a useful tool for estimating continuous distribution.

ACKNOWLEDGMENT

It is a great pleasure for us to express respect and deep sense of gratitude to our supervisor Dr. Dootika Vats, Associate Professor, Department of Mathematics and Statistics, Indian Institute of Technology Kanpur, as a convener of this course for her wisdom, vision, expertise, guidance, enthusiastic involvement and persistent encouragement during the planning and development of this work. We are truly grateful towards her for giving us a wonderful exposure to the topics of Non-Parametric Inference. We are also obliged to our parents for their moral support, love, encouragement and blessings to complete this task.

Akash Kumar Sharma
Arkonil Dhar
Souvik Bhattacharyya

Contents

1 Introduction

1.1 Context

In Bayesian classifier methods we have to adopt a suitable probability density function of the features conditioned on the class. Various possibilities are applicable, such the uniform, beta, Gaussian, etc. A simplified Bayesian classifier is the Naive Bayes that performs the classification task for any number of classes. It allows to predict the probability of affiliation of an element to a given class. It is a statistical classifier expanded by adding the so-called *naive* assumption that the input variables $x_1, x_2, \dots, x_N$ are independent of one another. As a result of this assumption, such a classifier deals well with any number of variables and together with the increase in the number of variables. Naive Bayes classifiers are easily implemented and highly scalable, with a linear computational complexity with respect to the number of data entries. The another bold assumption that Naive Bayes makes is that the numeric attributes are generated by a Gaussian distribution. Although this approximation may be true for many real world problems but this can not always be true. We need some other method to estimate the true density of the population. In this project our main focus is to estimate the true density of the population using kernel density estimation (KDE). The method of estimating the continuous density in Bayesian classifiers using KDE is called Flexible Naive Bayes. We next discuss some important properties of kernel density estimation that Flexible Bayes inherits. Later in this project we will compare the results of using Flexible Naive Bayes and the classical Naive Bayes on real world and simulated datasets.

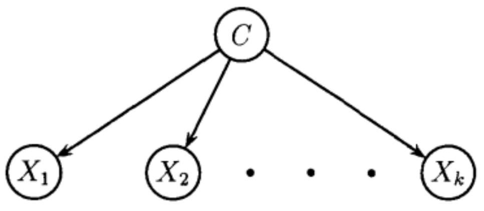

Figure 1: Figure 1: A naive Bayesian classifier depicted as a Bayesian network in which the predictive attributes $(x_1, x_2, ..., x_k)$ are conditionally independent given the class attribute (C)

1.2 Objectives of the Study

In this project,

1. We review the classical Naive Bayes algorithm.
2. We use Kernel Density Estimation technique to obtain an estimate of the feature vectors conditioned on a certain class and use it to create Flexible Bayes approach.
3. We check for the validity of the desired asymptotic property of Flexible Bayes.
4. We compare the two methods on both real world dataets and simulated datasets.

2 The Naive Bayesian Classifier

Naïve Bayes classifier is a probabilistic classifier based on Bayes' theorem, which assumes that each feature makes an independent and equal contribution to the target class. NB classifier assumes that each feature is independent and does not interact with each other, such that each feature independently and equally contributes to the probability of a sample to belong to a specific class. NB classifier is simple to implement and computationally fast and performs well on large datasets having high dimensionality. NB classifier is conducive for real-time applications and is not sensitive to noise. NB classifier processes the training dataset to calculate the class probabilities $P(y_i)$ and the conditional probabilities, which define the frequency of each feature value for a given class value divided by the frequency of instances with that class value. NB classifier best performs when correlated features are removed because correlated features will be voted twice in the model leading to the overemphasis of the importance of the correlated features. Naive Bayes algorithms are often used in sentiment analysis, spam filtering, recommendation systems, etc. They are quick and easy to implement but their biggest disadvantage is that the requirement of predictors to be independent. Let C be the random variable denoting the class of an instance and let X be a vector of random variables denoting the observed attribute values. Further, let c represent a particular class label, and let x represent a particular observed attribute value vector. Given a test case x to classify, we can use Bayes' rule to compute the probability of each class given the vector of observed values for the predictive attributes which is given as

$$P(C=c|X=x) = \frac{P(C=c)P(X=x|C=c)}{P(X=x)} \tag{1}$$

and then predicts the class for which probabibilty is maximum. Here $X = x$ represents the event that $X_1 = x_1, X_2 = x_2, ...X_k = x_k$ are independent and because these attributes are assumed to be conditionally independent, we have

$$P(X=x|C=c) = \prod_{i=1}^{k} P(X_i = x_i|C=c) \tag{2}$$

which is simple to compute for test cases and to estimate from training data. Also, the denominator in equation 1 is constant for each class we ignore the denominator and just compute the numerator in order to classify x to a class. We calculate this probability for each of the class and then assign x to that class which has the highest conditional posterior probability.

But when we have datasets that are not from Gaussian distributions, this classifier will not provide us the desired result for obvious reasons and this makes us look for a new approach of estimating the true densities without making any distributional assumption.

3 Kernel Density Estimation

Kernel Density Estimation is a non-parametric method to estimate the unknown probability density function of a random variable using the samples drawn from the random variable. We will use the Kernel Density Estimation technique to calculate the posterior distribution conditioned on a class in the Naive Bayes algorithm.

Let $x_1, x_2, ..., x_n$ be an i.i.d sample from some univariate distributions with unknown pdf f. Then the kernel density estimate of f is,

$$f_h(x) = \frac{1}{h}\sum_{i=1}^{n} K_h(x - x_i) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right),$$

where K is the *kernel*, a non-negative function, and $h > 0$ is the smoothing parameter termed as *bandwidth*. Popular kernel functions includes univariate, triangular, Gaussian, Epanechnikov etc. Also, the choice of bandwidth h can be different as we try to choose the optimum bandwidth using different criterion.

We can also extend this idea to the case of multivariate random variable. Let $\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_n}$ be d variate random vectors drawn from a common density f. In that case, the kernel density estimate will be given by,

$$\hat{f}_H(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} K_H(\mathbf{x} - \mathbf{x_i}),$$

where,

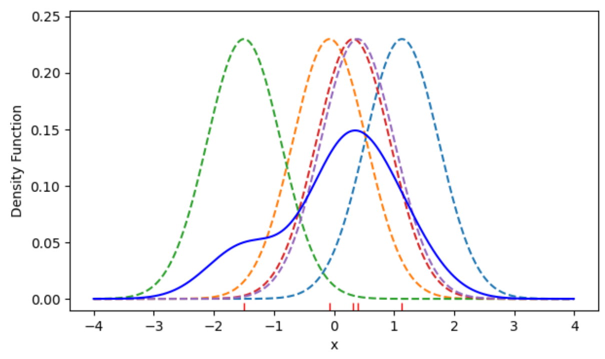

Figure 2: Kernel Density Estimate using 5 sample from $N(0,1)$ distrbution. The dashed curves are the individual kernels and the blue curve is the estimate. The data points are the rug plot on the x axis.

- $\mathbf{x} = (x_1, x_2, ..., x_d)$, $\mathbf{x_i} = (x_{i1}, x_{i2}, ..., x_{i3})$, i = 1,2,...n are d-vectors;
- H is the bandwidth (or smoothing) $d \times d$ matrix which is symmetric and positive definite;
- K is the kernel function which is a symmetric multivariate density and
- $K_H(\mathbf{x}) = |H|^{-1/2} K(H^{-1}\mathbf{x})$

3.1 Choice of Bandwidth

The bandwidth of the kernel is a free parameter which exhibits a strong influence on the resulting estimate. For example. for the univariate case, when we take the value of h to be too large, the value of kernel function at all the sample points become more or less equal and so the resulting estimate becomes too smooth. On the otherhand, when we take h to be very small, even smaller distance from the sample points gets amplified in side the kernel function and the final estimate becomes very squiggly.

For the multivariate case, we have to select the bandwidth matrix H, which establishes the degree of smoothing of the density function estimation. In this case, our goal is to select the bandwidth matrix H which is computationally inexpensive to compute. The number of parameters to be estimated for H, when working with d dimensional feature vectors is $d(d+1)/2$, and as d increases, the number of parameters increases as well. One choice of H could be the variance-covariance matrix matrix multiplied by some constant h, but in that case a desired asymptotic property (discussed in section 5) will not hold for the KDE and our model would fail to perform well. Keeping these in mind, and for the sake of simplicity, in this project we take H to be of the form $H = diag(h_1^2, ..., h_d^2) = h^2 diag(s_1^2, ..., s_d^2)$ where $s_i^2)$ is a dispersion measure (the sample standard deviation of X_i).

For selecting the value of h, we can follow two different approaches:

- Plug in method and
- Cross-validation method

The most common criterion for selecting the bandwidth is MISE or Mean Integrated Square Error, where $MISE(h) = \mathbb{E}[\int(\hat{f}_h(x) - f(x))^2]$. It can be shown that $h_{MISE} \propto n^{-1/5}$. In this project, we will be using 4 different bandwidths, namely: Scott's Factor, Silverman's Rule of Thumb Bandwidth and Cross Validation Maximum Likelihood and Cross Validation Least Square.

The Scott Factor is given by $h = n^{(-1/(d+4))}$, where n is the number of sample points and d is the the number of dimensions. For the multivariate case the Scott factor becomes

$$H = n^{-1/(d+4)} S,$$

where d is number of variables, n is sample size, S is the empirical covariance matrix. The bandwidth is returned as a covariance matrix, so to use it for a product kernel, we take square root of it's diagonal.

As the Silverman's rule of thumb bandwidth estimator, we have

$$h = 0.9 \min\left(\hat{\sigma}, \frac{IQR}{1.34},\right) n^{-\frac{1}{5}},$$

where $\hat{\sigma}$ is the standard deviation of the sample and IQR interquartile range and n denotes the sample size. For the multivariate case, we calculate $\hat{\sigma}$ and IQR for each of the feature variable and calculate h which serves the purpose of the corresponding diagonal element of the bandwidth matrix H. The CV-LS and CV-ML is based on a cross-validatory approach to find the value of optimum h which is done by using leave-one-out kde.

3.2 Choice of Kernel

A kernel function K has three features:

- K is non- negative.
- K is symmetric.
- $\int K(x)dx = 1$

As we can imagine, the second requirement is needed to make sure that the kde is a valid density function. We also note that, as the number of increases the choice of kernel does not play an important role in the estimation of true density. Through out this project, we will use the Gaussian Kernel for our estimation purpose. Below we consider three most common kernel functions and apply them to the faithful dataset:

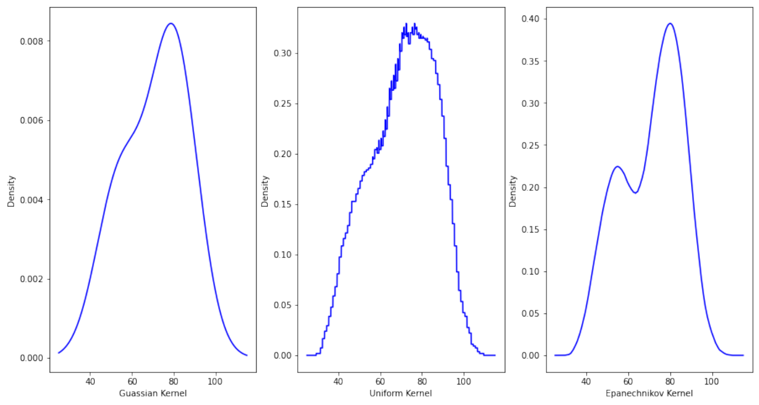

Figure 3:

The form of the three kernels used above is:
Guassian Kernel: $K(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}$,
Uniform Kernel: $K(x) = \frac{1}{2} I(-1 \le x \le 1)$,
Epanechnikov Kernel: $K(x) = \frac{3}{4} \cdot \max(1 - x^2, 0)$.

There are other kernels such as triangular kernel, biweight kernel, cosine kernel etc. In this project we will use only the standard Gaussian Kernel.

4 Flexible Bayes Classifier

Now we introduce the Flexible Bayes learning algorithm, which is exactly the same as Naive Bayes in all respects but one: the method used for density estimation on continuous attributes. Here we will follow the non parametric route by estimating the class conditional densities by using Kernel Density Estimation. So, we will replace the expression in equation 2 with our kde estimate. The rest of the method remains same. The estimated density would be of the form,

$$P(X = x \mid C = c) = \frac{1}{n} \sum_i g(x, x_i, h),$$

where g is the Gaussian Kernel under consideration. In Naive Bayes, for an unclassified new observation, we have to compute the the posterior probabilities which only requires the μ vector and σ for each of the different classes. But in the Flexible Bayes approach we need to store all the training observations separately for the kernel density estimation purpose. This leads to some increased time and space complexity when compared to the classical naive bayes approach. This drawback is summarized in the following table.

Opertaion	Naive Bayes Classifier		Flexible Bayes Classifier	
	Time	Space	Time	Space
Train on n cases	O(nd)	O(d)	O(nd)	O(nd)
Test on m cases	O(md)		O(mnd)	

Table 1: Algorithmic complexity for Naive Bayes and Flexible Bayes, given n training cases, m test cases and d features.

5 Asymptotic properties of Flexible Bayes

In this section, we will define the strong pointwise consistency property and show that Flexible Bayes model do posses this desired asymptotic property.

5.1 Definition

If f is a probability density function and $\hat{f}$ is an estimate of f based on n examples then then $\hat{f}$ is strongly pointwise consistent if $\hat{f} \to f(x)$ almost surely for all x; i.e. for every ε,

$$p\left(\lim_{n\to\infty} |\hat{f}(x) - f(x)| < \varepsilon\right) = 1$$

Now we will prove the strong consistency of Flexible Bayes.

5.2 Theorem 1 (Strong Consistency for Nominals):

Let $X_1, X_2, \ldots, X_n$ be an independent sample from a multinomial distribution with v values, where the probability of drawing value j is p_j. Let $n_j = \sum_i I(X_i = j)$, the number of samples with value j. Then $\frac{n_j}{n}$ is a strongly consistent estimator of p_j.

Proof: n_j is the number of samples with value j, i.e $n_j = \sum_i I(X_i = j)$. Here $I(X_i = j)$ is a Bernoulli random variable with p_j as the probability of success. Hence, $I(X_1 = j), I(X_2 = j), \ldots, I(X_n = j)$ can be thought of as a sequence of i.i.d Bernoulli random variables with mean $E(I(X_i = j)) = p_j$ and standard deviation $\sqrt{p_j(1-p_j)}$. By using the *The Strong Law of Large Number*, we have $P\left(\lim_{n\to\infty} \frac{\sum_i I(X_i=j)}{n} = p_j\right) = P\left(\lim_{n\to\infty} \frac{n_j}{n} = p_j\right) = 1$ ■

5.3 Theorem 2 (Strong Consistency for Reals):

Due to Devroye (1983), the kernel density estimate is strongly consistent when:

- The kernel function must be a bona fide density estimate - it must be non-negative for all x, and it must integrate to 1.
- $h_n \to 0$ as $n \to \infty$
- $nh_n \to \infty$ as $n \to \infty$

5.4 Lemma 1 (Consistency of Products):

Let the functions $\hat{f}_1, \hat{f}_2, ..., \hat{f}_n$ be strongly consistent estimates of density functions $f_1, f_2, ..., f_k$. Then $\prod_i \hat{f}_i$ is a strongly consistent estimator of $\prod_i f_i$.

***Proof*:** If we can prove that $\hat{f}_1 \hat{f}_2$ is strongly consistent for $f_1 f_2$, then the generalization can be done via mathematical induction.

We need to show that, for all x,

$$P\left(\lim_{n\to\infty} |\hat{f}_{1,n}\hat{f}_{2,n} - f_1 f_2| < \varepsilon\right) = 1 \text{ for any preassigned } \varepsilon.$$

Also, since $\hat{f}_1$ and $\hat{f}_2$ are strongly consistent for f_1 and f_2, we have for any $\varepsilon_1, \varepsilon_2$

$$P\left(\lim_{n\to\infty} |\hat{f}_{1,n} - f_1| < \varepsilon_1\right) = 1 \text{ and } P\left(\lim_{n\to\infty} |\hat{f}_{2,n} - f_2| < \varepsilon_2\right) = 1$$

Now,

$$\begin{aligned}|\hat{f}_{1,n}\hat{f}_{2,n} - f_1 f_2| &= |(\hat{f}_{1,n}\hat{f}_{2,n} - f_1\hat{f}_{2,n}) - (f_2\hat{f}_{1,n} - \hat{f}_{1,n}\hat{f}_{2,n}) - (\hat{f}_{1,n}\hat{f}_{2,n} - f_1\hat{f}_{2,n} - f_2\hat{f}_{1,n} + f_1 f_2)| \\ &\le \hat{f}_{2,n}|\hat{f}_{1,n} - f_1| + \hat{f}_{1,n}|\hat{f}_{2,n} - f_2| + |\hat{f}_{1,n} - f_1||\hat{f}_{2,n} - f_2| \\ &\le \hat{f}_{2,n}\varepsilon_1 + \hat{f}_{1,n}\varepsilon_2 + \varepsilon_1\varepsilon_2, \text{ when } n \to \infty\end{aligned}$$

Since the above expression holds true for arbtrary ε_1 and ε_2 and $\hat{f}_{1,n}$ and $\hat{f}_{2,n}$ are finite, we can make $\hat{f}_{2,n}\varepsilon_1 + \hat{f}_{1,n}\varepsilon_2 + \varepsilon_1\varepsilon_2$ arbitrarily small so that the bound on ε can be made to hold. ■

5.5 Theorem 3 (Consistency of Flexible Bayes):

Let the true conditional distribution of the class given the attributes be $P(C|X) = \frac{\prod_i P(X_i = x_i|C=c)P(C)}{\prod_i P(X_i = x_i)}$. Then the Flexible Bayes estimator $\hat{P}(C = c|X = x)$ is a strongly consistent estimator of $P(C = c|X = x)$

***Proof*:** By Theorems 1 and 2, Flexible Bayes' estimates of $P(X|C), P(X)$ and $P(C)$ are strongly consistent. Hence using Lemma 1 we have Flexible Bayes' estimate of $P(C|X)$ is strongly consistent. ■

6 Simulation and Data Analysis

In this section we will provide a comparative study of simulation and data analysis done using Naive Bayes and Flexible Bayes.

In simulation studies, we considered one Gaussian and one non-Gaussian populations. Data are generated from a multivariate Gaussian distribution where the location parameter is $(1,2,3,4,5)'$ and scale parameter is 1 with

- Number of classes is 5.
- Class sizes are respectively 100, 110, 95, 101 and 99.

Also data are generated from a multivariate Laplace distribution with the location parameter is $(1,2,3,4,5)'$ and scale parameter 1 with

Dataset	Size	#Class	Naive Bayes	Flex (Scott)	Flex (Silverman)	Flex (CV-LS)	Flex (CV-ML)
Normal Population	505	5	0.8237	0.9801	0.9845	0.9713	0.9581
			0.8823	0.8431	0.8627	0.7647	0.8039
Non-Normal Population	505	5	0.7224	0.9647	0.9735	0.9669	0.9339
			0.6274	0.6274	0.647	0.549	0.6862

Table 2: The training and test accuracies are reported for the corresponding models. As can be seen from above, Flexible Bayes classifier outperforms Naive Bayes classifier in most of the cases.

- Number of classes is 5.
- Class sizes are respectively 100, 110, 95, 101 and 99.

For real life data, the Iris dataset, Wisconsin Breast cancer dataset and the Wine dataset are used. The size, number of classes and the respective training and test accuracies for the models are reported below.

Dataset	Size	#Class	Naïve Bayes	Flex (Scott)	Flex (Silverman)	Flex (CV-LS)	Flex (CV-ML)
Wisconsin Breast Cancer	505	2	0.9342	0.9953	0.9976	1	0.9765
			0.9301	0.937	0.951	0.4196	0.944
Iris Dataset	505	3	0.9464	1	1	1	0.991
			1	1	1	0.7894	0.9136
Wine Dataset	178	3	0.9924	1	1	0.9849	1
			0.9333	1	1	0.6444	0.9778

Table 3: The training and test accuracies are reported for the corresponding models for the real data sets. As can be seen from above, Flexible Bayes classifier outperforms Naive Bayes classifier in most of the cases.

7 Conclusion

In this project we have reviewed the classical Naive Bayes approach and its assumptions of Normality of the feature vectors. Then we suggested the Flexible Bayes classifier which uses Kernel Density Estimation to approximate the population density which proves to be an improvement over the classical approach when the actual data are not from Gaussian distribution. We have also discussed about the strong point-wise consistency property which Flexible Bayes method possess. Then we applied both the models on real world and simulated data to get the desired results.

8 Appendix

The codes used for simulation and data analysis part is available here.

References

[1] A new Bayesian procedure for testing point null hypotheses: Yuliang Yin

[2] Cross-validation MLE Fitting of Kernel Density Estimator, With Variety of Kernels

[3] The L_1 Convergence of Kernel Density Estimates: Luc Devroye, T. J. Wagner

[4] Kernel Density Estimation By: Matthew Conlen.

[5] Intro to Kernel Density Estimation.

[6] Wikipedia