Comparison of Naïve Bayes classifier and Bayes' classifier with K nearest neighbor method

EE 140 - Stochastic Processes, Detection, and Estimation
Department of Electrical and Computer Engineering
Tufts University Fall 2017
Term Project

Jiacheng Qu
Dec. 22, 2017

Introduction

In the proposal for this project, I proposed the idea of working on existing methods on the topic of classifications including but not limited to locally trained piecewise linear classifiers, K-nearest neighbors (KNN), Gradient boosting, Support vector machines (SVM), Decision trees such as Classification and Regression Tree (CART). In the past two decades, Classification has become an important aspect not only on the topic of data analysis but also on the topic of image processing. However, after several meetings with Prof. Miller, I noticed that the requirement of including probabilistic and stochastic concepts in the paper is mandatory. Besides, I found that to work on topics mentioned above could be a bit challenging given the time and knowledge of myself. It consists of lots of pre-existing expertise which could be acquired both in and out of class. After another turn of research online and the list of suggested topics, I found Bayes classifier with probability densities estimated and the k nearest neighbor method (KNN) with Bayes' theorem are those of my best interests. For the implementation section, I chose to move on to the Bayes classifier and came up with a preliminarily design/idea: investigate and implement Bayes classifier; besides, utilizing Bayes-KNN and other comparable approaches could be a useful/helpful reference, but not necessarily.

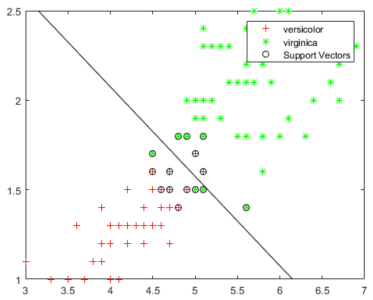


Figure 1 Classification of 2-dimensional data with two features.
Horizontal axes denote feature 1and vertical axes denotes feature 2.

For each question in machine learning, there are two major categories, regression, and classification. In these two issues, thousands of papers and analyses have been made throughout the decades. In a regression problem, prediction of the results has been made among a set of consecutive numbers. While in a classification problem, researchers would utilize discrete methods to predict/ decide on the categories of its output [8]. However, I would like to draw on the latter one, and this project may also cover material from the lectures, such as Bayesian method, posterior probability distribution, and decision theory.

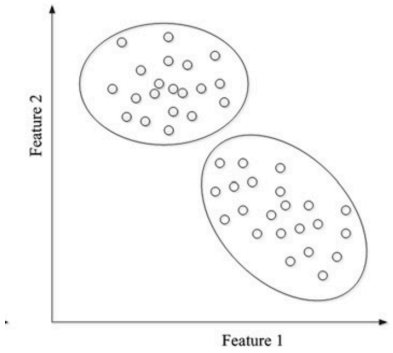


Figure 2 Clustering of two feature data inputs

Figure 2, the idea of clustering is another way of defining classes other than classification, where class boundaries are arithmetically calculated but not sharply defined [11]. However, as indicated by Figure 3, it is more likely to have encounter zones between two or more classes of feature sets and this is also the case being discussed and analyzed in this paper [10].

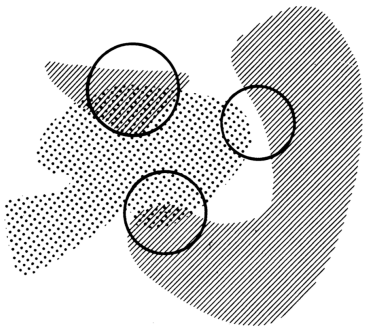


Figure 3 Encounter zones among mutliple classes of featured data

Before move on to the main sections (approaches and experiment) of this project, I found the couple of useful built-in Matlab functions such as classify(), svmtrain() and predict() by chance. Especially, predict() is a function that could visualize posterior classification probabilities predicted by a naive Bayes classification model [6].

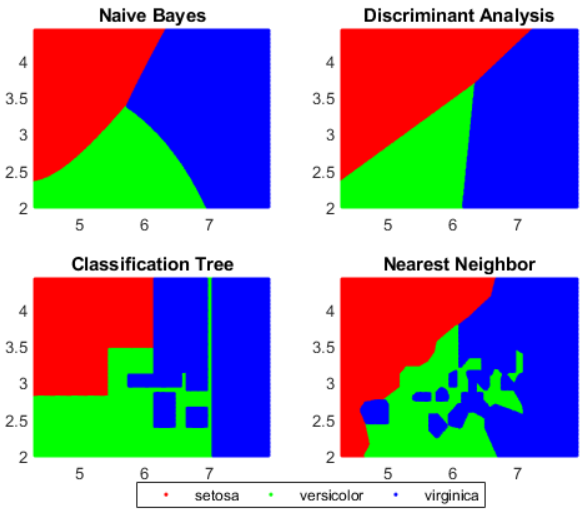


Figure 4 Visualize Decision Surfaces of four classifiers using Matlab built-in functions

Figure 4 shows the plots of Visualize Decision Surfaces of four built-in classifiers with four different decision-making rules. For instance, the upper left graph is a result of locally trained naive Bayes classifier by type in the command: mdl = fitcnb(X, Y). These built-in functions gave me the general picture of different existing and efficient methods for classification, and it also inspired me about the hands-on experiment that I am going to build in the paper.

Also, other than implementations of classification methods, the team found that testing and verification are necessary for not only the project completeness but also the functional completeness. However, exploit testing, and verification methods could be challenging and time-consuming.

Index Term

adjacency matrix, bates decision surface, classifier, pattern recognition, trained naïve bayers classifier, computational speed memory requirement, error rate, minimum square root error, maximum likelihood classification, piecewise linear classification, posterior probability distribution

Objectives

The expected audience of this paper is engineering entrance level students or even novices. This article should deliver a bright idea of the following:

- Naïve Bayes Probabilistic Classifier/Posterior classification probabilities
- Bayes classifier assuming Gaussian distributions
- Bayes classifier with probability densities estimated
- Bayes classifier with the K nearest neighbor method (B-kNN)

Approaches

1) *Naïve Bayes Probabilistic Classifier*

As recognized as part of the family of statistical classifiers Naïve Bayes Probabilistic Classifier is based on Bayes theorem with assumptions that features are naive independent between each other, and the general decision rule of a Naïve Bayes Probabilistic Classifier has been stated below [1]:

$$P(C_i|X) > P(C_j|X)$$

Bayes Theorem:

$$P(k/x)P(x) = P(x/k)P(k)$$

$$\text{where} \quad P(x) = \sum_{k=1}^{K} P(x/k)P(k)$$

P(k) = a priori probability of class k

P(k/x) = a posteriori probability of class k given the unknown input x

P(x/k) = class conditional pdf of x given the class info k

where k represents the different groups/classes of trained data

in other words, Posterior = prior*likelihood/evidence

Therefore, the predicted class (y) is defined as below:

$$y = \arg \min_{y=1,..K} \sum_{k=1}^{K} P(k/x)C(y/k)$$

In the flowing of this paper, the cost of classification C(y/k) would be simplified by using constant number (C(k/k) = 0 while C(`k/k) = 1) instead of reducing the complexity of the calculation; i.e, the cost of classifying class k to its true class is 0 while an inaccurate classification would have unit cost [6].

$$\log \frac{p(C_1 \mid \mathbf{x})}{p(C_2 \mid \mathbf{x})} = \log p(C_1 \mid \mathbf{x}) - \log p(C_2 \mid \mathbf{x}) > 0$$

From the in-class material, the Likelihood-ratio test could be used for designing the decision rule. By the naïve Bayes classifier, estimation of the densities of predictors for each class could be done together with modeling posterior probabilities based on the Bayes rule.

Bayes classifier using Quad-tree/histogram assuming Gaussian distributions

While many of the paper on the topic of classifier stated that the probability density of each class and mean of each sub-classes are known, here I am going to estimate the pdf for each class. To generate the class conditional pdf P(x/k) of x given that the class is k, two methods have been undertaking here.

- Quadtree/Octree Data Structures

On the one hand, Quadtree/Octree Data Structures have been used for weighting the pdf of x given the class info k.

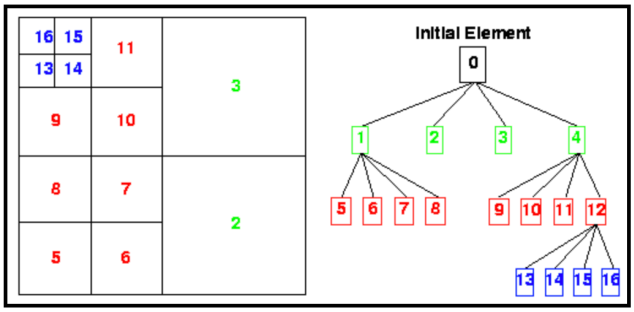


Figure 5 Quadtree/Octree data structures

From the left-hand side of the graph, region number 5, 6, 7 and 8 belong to 1 while 12, 14, 15 and 16 belong to region number 12. The tree diagram on the right forthrightly shows the data structure [9]. Assigning the space is relatively simple, slides the square cell into four pieces while there is at least one data point inside that cell; otherwise, stop cutting. After assigning all of the data points into individual cells, start counting from top-down, every time the tree branch goes down by one level, add unit value to the corresponding weight of that cell/data point.

- Histogram *with Gaussian filter*

On the other hand, by assuming Gaussian distributions, the pdf of x given the class info k is calculated by plotting the histogram and smoothing using a Gaussian filter.

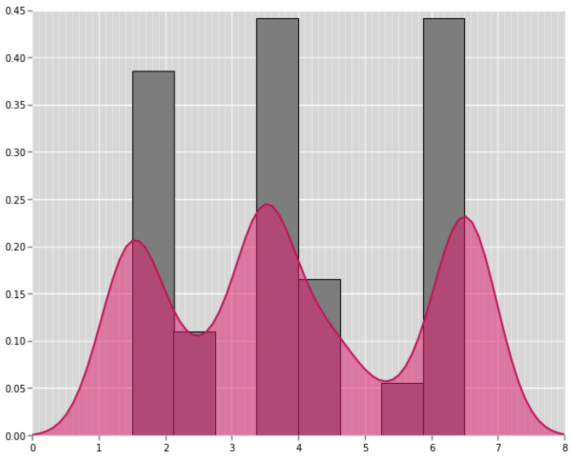


Figure 6 Smoothing histogram by applying Gaussian filter

Plotting histogram of one class of one of two features, the distribution could be calculated as stated in figure 6 [6]. By assumption, the distribution is a Gaussian distribution; thus, applying Gaussian filter would ensure every possible data range has positive pdf value. Repeat for the other side of the two features for the same class and multiply the two vectors by each other for a matrix of pdf of that particular class. Similarly, the pdf of all existing classes could be estimated [4].

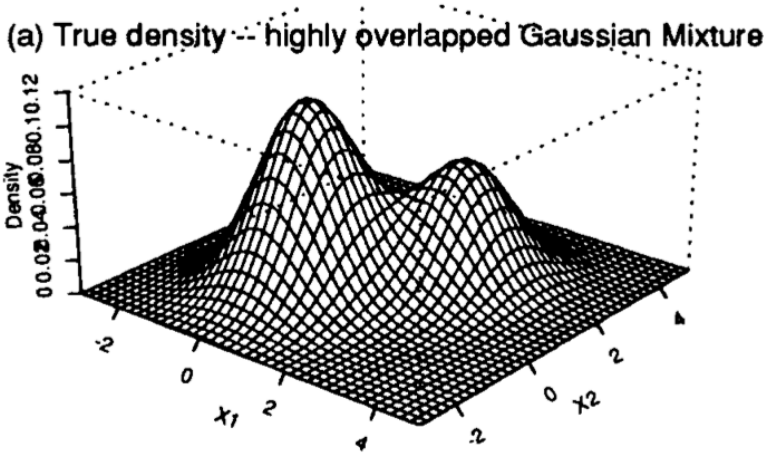


Figure 7 Nonparametric multivariate density estimation

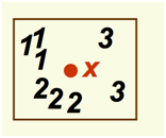
2) Bayes classifier with probability densities estimated with the KNN method

Based on the preliminary research, I decided to figure out the details of implementation of Bayes-KNN method

- *KNN from a Bayesian viewpoint*

The basic idea of KNN algorithm is to find out k nearest neighbor of the target spot for

each class.



Using the same method for the estimation of the posterior in Naïve Bayes Classifier, posterior here could be estimated by [5]:

$$P(c_i \mid x) = p(x, c_i)/p(x) = k_i / \sum(k_j) = k_i/k$$

- *Advanced K nearest neighbor method*

$$w_i = \begin{cases} \frac{d_k^{NN} - d_i^{NN}}{d_k^{NN} - d_1^{NN}} & : d_k^{NN} \neq d_1^{NN} \\ 1 & : d_k^{NN} = d_1^{NN} \end{cases}, \quad i = 1, ..., k$$

$$p_j = \frac{\sum_{i=1}^{k} w_i \cdot I(c_j = c_i^{NN})}{\sum_{i=1}^{k} w_i}, \quad j = 1, ..., l$$

The probability of x belongs to class j is equal to p_j while w_i denotes the weight of i-th nearest point to the target spot [2].

$$c_r = \arg\max_{c_j} P, \quad j = 1, ..., l$$

where c_r is the resulting class.

Experiments

For the input data, I pulled out the data sets that I measured from my master project. To be specific, it is a set of data of audio signal that similar to the Matlab sample data: Fisher's iris data set.

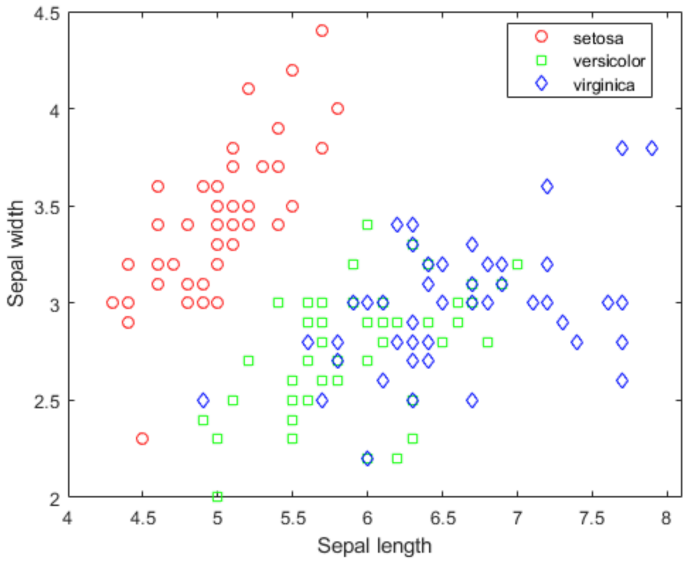


Figure 8 Matlab sample data: Fisher's iris data

Following is the feature space of five different classes of data: rail noise with horns, rail noise without horns, aircraft noise, truck noise, and MBTA buses noise.

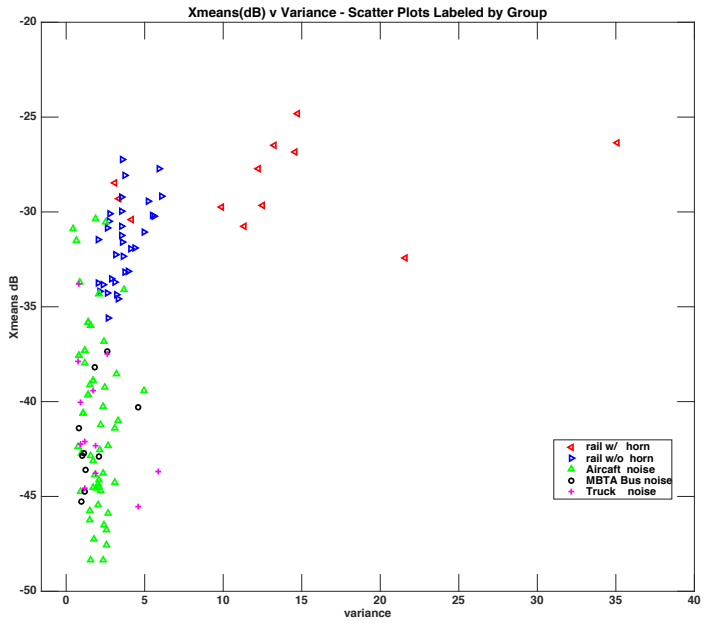


Figure 9 Master Project's data set

To compare classifiers mentioned above, an analysis of the selected outcomes need to be made. Here, I circled ten data points using a purple ring. In each step, I removed one of them from the input data set and put it back for the testing purpose:

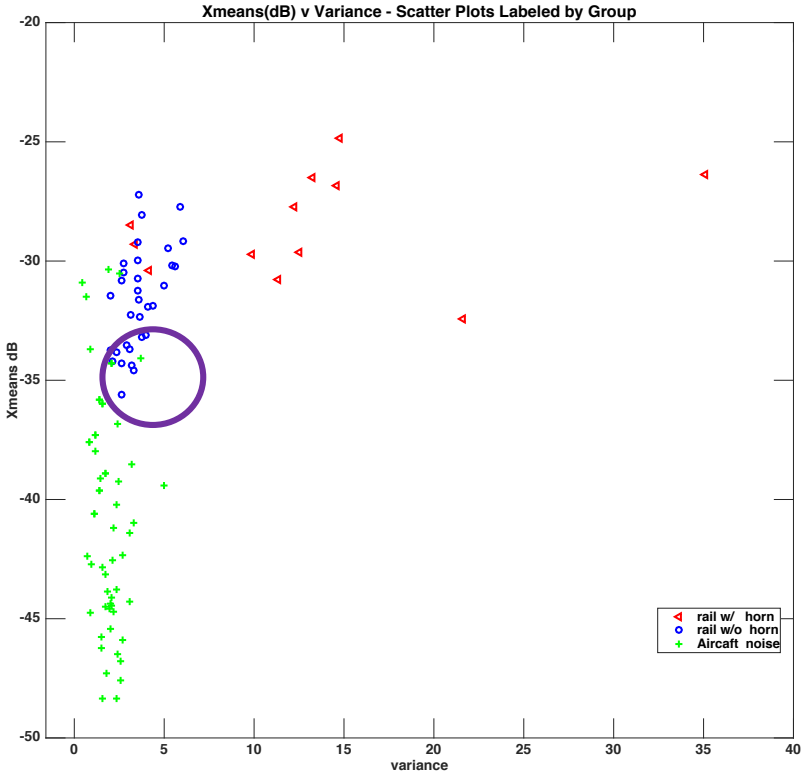


Figure 10 rail noise with horns, rail noise without horns, aircraft noise

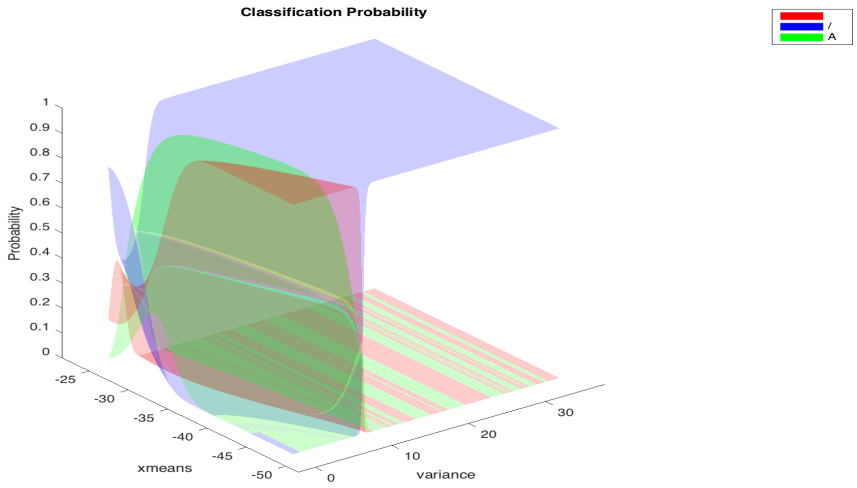


Figure 11 Naive Bayes with pdf estimated

Table 1 True class vs. the guesses

True class:	2 Green			8 Blue		
Bayes with pdf	1/2			8/8		
Bayes-KNN	0/2	0/2	0/2	7/8	7/8	8/8
B-KNN with pdf estimated	0/2	0/2	0/2	7/8	8/8	8/8
	K=5	K=7	K=9	K=5	K=7	K=9

Conclusion

The goal of this project is to discover and analysis existing classification methods, testing simple probabilistic methods in machine learning including Bayes' classification, Bayes-KNN algorithm has been briefly discussed as planned. In conclusion, by using the approaches above, straightforward Naïve Bayes classifier by pdf estimation p(x) does not work very well with kNN approach because the resulting density estimate needs lots of data input as training data, and some of the data selected for testing were too isolated from their classes which made the entire trial a particular case. After optimizing the B-kNN algorithm by pdf estimation, I found that the accuracy of the new method is better than the naïve Bayes one, but it is still relatively rough. It is also worth mentioning that the classification of two isolated points (green) didn't get any improvement at all due to the boundary condition. To optimize the result further, we need to set more data points. It is worth mentioning that such a promotion could be a result of the addition of weighted distance w_i.

Moreover, it would be meaningful for us to implement an algorithm that can set these training points continuously and accurately. However, given the limited time and knowledge, the team has had some trouble with the notion of completeness. If I were to pursue this topic, I would work more on this issue to make the algorithm more supervised and set data features correctly. Furthermore, we will use more data points such that we can get a more precise result.

Bibliography

1. Anwar, S. M., Saeed, S. M., & Majid, M. (2016). Classification of Expert-Novice Level of Mobile Game Players Using Electroencephalography. *2016 International Conference on Frontiers of Information Technology (FIT)*. doi:10.1109/fit.2016.064
2. Chatzigeorgakidis, G., Karagiorgou, S., Athanasiou, S., & Skiadopoulos, S. (2015). A MapReduce based k-NN joins probabilistic classifier. *2015 IEEE International Conference on Big Data (Big Data)*. doi:10.1109/bigdata.2015.7363844
3. Hand, D. J. (1996). *Discrimination and classification*. Chichester: Wiley.
4. Hwang, J., Lay, S., & Lippman, A. (1994). Nonparametric multivariate density estimation: a comparative study. *IEEE Transactions on Signal Processing, 42*(10), 2795-2810. doi:10.1109/78.324744
5. Liu, Y., Chen, Y., & Cheng, J. (2016). Feature extraction of protein secondary structure using 2D convolutional neural network. *2016 9th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*. doi:10.1109/cisp-bmei.2016.7853004
6. MATLAB 2017b, The MathWorks, Natick, 2017 Retrieved December 25, 2017, from https://www.mathworks.com/help/stats/visualize-decision-surfaces-for-different-classifiers.html
7. Milkbox.net. (n.d.). Gaussian kde smoothed histograms with matplotlib. Retrieved December 28, 2017, from http://milkbox.net/note/gaussian-kde-smoothed-histograms-with-matplotlib/
8. Ng, A. (2017). Machine Learning. Retrieved December 27, 2017, from https://www.coursera.org/learn/machine-learning
9. Quadtree/Octree Data Structures. (n.d.). Retrieved December 28, 2017, from http://www.cs.sandia.gov/~kddevin/LB/quadtree.html
10. Sklansky, J., & Michelotti, L. (1980). Locally Trained Piecewise Linear Classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI-2*(2), 101-111. doi:10.1109/tpami.1980.4766988
11. Suthaharan, S. (2016). *Machine learning models and algorithms for big data classification: thinking with examples for effective learning*. New York: Springer.