EE5907 CA1 report
Yang Di A0229522H

Q1. Beta-binomial Naive Bayes

Briefly introduction of theorem:

Among all machine learning classification algorithms, Naive Bayes is different from most other classification algorithms. For most classification algorithms, such as decision tree, KNN, support vector machine, etc., they are discriminant methods and directly learn the relationship between output Y and feature X, either decision function $Y = f(x)$ or conditional distribution $P(Y|X)$. But Naive Bayes is to directly find the joint distribution of the feature output Y and feature X, and then use the $P(Y|X) = P(X,Y)/P(X)$ to derive it. And the basic Bayes function is $P(Y|X) = P(X|Y) * P(Y)/P(X)$, in other terms Y refers to class, X refers to feature. Finally, we just need to calculate $P(Class|Feature)$ to make a point.

As for beta-binomial distribution, it is a kind of discrete probability distribution function of finite space values in probability theory and statistics. It differs from the general binomial distribution. Although it also represents the probability of success of a series of known Bernoulli experiments, the Bernoulli constant will become a random variable.

Results and discussion:

Figure. 1 Training and Testing Error rates versus various alpha. (Threshold of binarization is 0).

a	1	10	100
Training data Error rate	11.55%	12.01%	14.19%
Testing data Error rate	11.59%	11.91%	13.67%

According to figure.1, it is obvious that with a increase, the Error rates for both training data and testing data trend to decrease. What's weird here is that the training set error rate is slightly higher than that of testing set while a is larger than 10 approximately. So as for the three a in the table, 1 should be an optimized parameter.

Even though the topic requires the threshold value of binarization at 0, I find that slightly increase the threshold value to like 0.2, can solve the weird phenom mentioned before, and the error rate can be reduced, as shown in figure.2

Figure. 2 Training and Testing Error rates versus various alpha. (Threshold of binarization is 0.2).

Q2. Gaussian Naive Bayes

Briefly introduction of theorem:

As mentioned before, I have briefly introduced Naïve Bayes, so I just make some brief introduction on Gaussian distribution. If feature X is a continuous variable, how to estimate the likelihood of $P\ (X|Y_i)$? We can assume that x follows a Gaussian (alson known as normal) distribution in the case of Y_i. According to the normal distribution of probability density function to calculate the $P\ (X|Y_i)$, the formula is as follows:

$$P(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-u)^2}{2\sigma^2}}$$

Results and discussion:

Training data Error rate	16.57%
Testing data Error rate	16.02%

Q3. Logistic regression

Briefly introduction of theorem:

Generally speaking, Logistic Regression is a machine learning method for solving dichotomies (0 or 1) that estimate the likelihood of something. For example, the likelihood that a user will buy a product, the likelihood that a patient will have a disease, And "possibility" is used here, not the mathematical "probability". The result of logistic regression is not the probability value in the mathematical definition, so it cannot be directly used as the probability value. This result is often used to add weights to other eigenvalues rather than multiply them directly. Then I will introduce several important points about logistic regression.

As for Hypothesis function sigmoid function (Logistic function):$g(z) = \frac{1}{1+e^{-z}}$.

The sigmoid function is an S-shaped curve with values between [0, 1]. The value of the function quickly approaches 0 or 1 away from 0. As for decision boundary, it's a plane or a surface used to separate different classes of samples in n-dimensional space. The decision boundary is actually a function defined by $\theta^T x = 0$ in logistic regression. So $P(Y = 1|X;\theta) = g(\theta^T x) = \frac{1}{1+e^{-\theta^T x}}$. As for cost function, Generally speaking, any function that can measure the difference between the model predicted value $h(\theta)$ and the real value y can be called the cost function. If there are multiple samples, the value of all the cost functions can be averaged and denoted as $J(\theta)$. In linear regression, the most commonly used is Mean squared error. However, in logistic regression, the most commonly used cost function is cross entropy

Results and discussion:

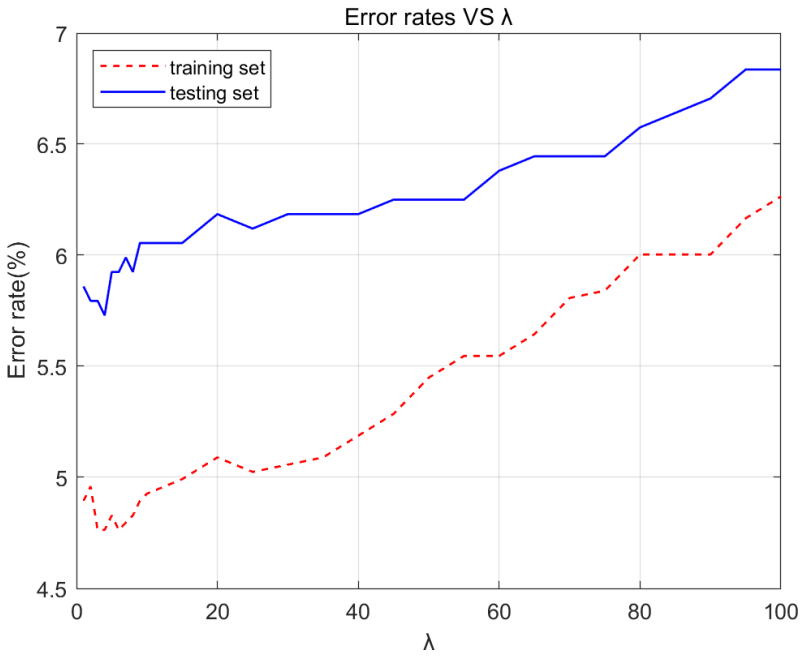


Figure. 3 Training and Testing Error rates versus various lambda.

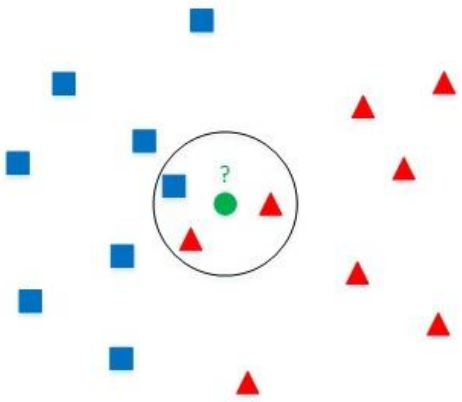
λ	1	10	100
Training data Error rate	4.89%	4.93%	6.26%
Testing data Error rate	5.86%	6.05%	6.84%

As shown in figure. 3, when λ increases, the error rate also increases. Base on the chosen three parameter, $\lambda = 1$ should be an optimized one.

Q4. K-Nearest Neighbors

Briefly introduction of theorem:

Given a training data set, k-nearest neighbor algorithm finds K instances closest to the new input instance in the training data set. Most of these K instances belong to a certain class, and then classifies the input instance into this class.



As shown in the figure above, there are two different types of sample data, which are represented by small blue squares and small red triangles respectively. The data indicated by the green circle in the middle of the figure is the data which need to be classified. According to the idea of k-nearest neighbors. Example given here, If K=3, the three nearest points to the green dot consist of two small red triangles and one small blue square. Based on the statistical method, the green point is classified to belong to the red. If K=5, based on statistical methods, the green point to be classified belongs to the blue.

Results and discussion:

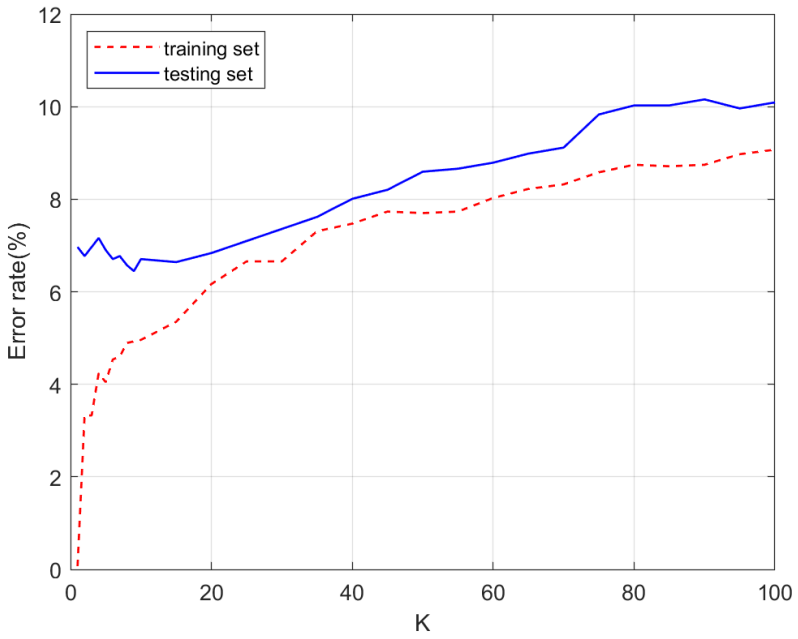


Figure. 4 Training and Testing Error rates versus various K.

K	1	10	100
Training data Error rate	0.00065%	4.96%	9.07%
Testing data Error rate	6.97%	6.71%	10.09%

As shown in figure. 4, for train data, with K increase, the error rate increases. When K=1, the error rate almost approaches 0, as the features that need to predict can find the exactly same features in the training set. Actually, there is a bug for the program, if there are more than K number of same smallest distance, the system will pick the point which has smaller label, which will influence the predictive result. It is why when K=1, the error rate of train data is very close to 0 instead of 0. For test data, the error rate firstly decreases slightly then increases, the lowest point can be found when K=7.

Q5.

Honestly speaking, I almost spent 2 weeks on this assignment. I find that just using for loop nesting to compute Euclidean distance is time wasting. Then I change my way to use matrix computation which save a lot of time.