MACHINE LEARNING ASSIGNMENT 2

PRATEEK AGARWAL – A20358932

Table of Contents

Problem Statement

We are trying the solve the problem of classification using the Generative learning approach. Classification is the problem of identifying a class for a given feature vector. We use different dataset to train our machine for this task. To do this we assume that each data set has different distribution for example Gaussian, Bernoulli's and Binomial. We test assuming all the distribution and find different measures such as Recall, Precision, Accuracy and F measure to evaluated our algorithm.

Following the are five types of data set we take as examples and build algorithm assuming different distribution.

- One dimension Two Class Gaussian Discriminant Analysis
- N Dimension Two Class Gaussian Discriminant Analysis
- N Dimension K Class Gaussian Discriminant Analysis
- Naive Bayes with Bernoulli's Distribution
- Naïve Bayes with Binomial Distribution

Proposed Solution

For the first three cases we take the following assumptions in account.

- All the features are continuous in nature.
- Examples are independent.
- All the features belonging to same class have a Gaussian distribution (Univariate in case of one dimension and Multivariate in case of n dimension.
- Concluding from the last point we get that each class have a prior class probability called alpha(α), Variance called ($\sum$) and mean (μ).

So, what we have now is a X matrix with continuous features and a Y vector denoting the labels of each features. Our aim to find with what probability a new feature vector will have a particular class. For example, if X (a new feature vector) has more probability of being in class 1 then class 0 then it is assigned class 0. What we have in probability of a feature vector for a given class. So, we use Bayes Rule.

$$P(Y=j/X) = (P(X/Y=j) * P(Y=j)) / P(X)$$

We estimate the following parameter of each class (this is called training part).

M_j = number of examples of class j.

μ_j = mean of class j.

$\sum_j$ = Covariance of class j.

Next, we use the formulae of Multivariate distribution to calculate probability of a given X in a particular class. Let's call this membership ship function g(X). More the membership of a particular class more the X belongs to that class.

10 cross fold can be used to validate how well is this working.

In case we are not sure of the distribution the feature vectors are following, we can use a likelihood function and maximize it to get to get the membership function.

$$L(\Theta) = P(X^1, X^2, \ldots\ldots\ldots\ldots, X^m / \Theta)$$

We know that all the examples are independent, therefore.

$$P(X^{1,} X^2 \ldots\ldots\ldots, Xm) = P(X^1) * P(X^2) * \ldots\ldots\ldots\ldots * P(X^m)$$

We use the log likelihood on the above function and maximized it by taking its gradient and equating it to zero. We shall get the same result if we plugin in multivariate distribution on the log likelihood approach.

For the last two cases we take the following assumptions:

- Not only the examples are independent but each feature in an examples is also independent.
- For Bernoulli's distribution we assume all the features are binary i.e. they exist or they do not exist in an example.
- For Binomial distribution we assume the features are discrete in each example.
- Each class has a prior class

So, what we have now is a X matrix with Binary. Discrete features and a Y vector denoting the labels of each features. Our aim to find with what probability a new feature vector will have a particular class. For example, if X (a new feature vector) has more probability of being in class 1 then class 0 then it is assigned class 0. Each class has a prior class probability and each feature has a alpha (α) for belonging to a particular class.

We use the same approach as before of maximizing the log likelihood to find the parameter alpha for each feature and then use this to find the probability of a new feature belonging to a class. Laplace smoothing to used to remove the error that occurs when a feature a value of zero through out. It is done by adding a small number while calculating the alpha parameter.

Implementation details

Algorithms discussed above is implemented for five example data set. Several functions are build and used according to their functionality.

- **Indicator function** returns 1 if value of the label and the class passed are same else returns 0.
- **Cal_mu, Cal_alpha, Cal_alpha** calculates the parameter μ, $\sum$ and α for given class
- **Training** function creates a list of μ, $\sum$ and α for all the class.
- **Membership** function calculates the membership of a feature belonging to the given class.
- **Measures** function calculate parameters to evaluate our algorithm such as recall, precision, accuracy and F measure.
- **ROC_curve fintion** takes X,Y, cat and threshold and calculates the precision and recall for the given threshold.

Data set used are as follows:

- For N dimension two class we use the Bank Note Authentication available at UCI data repository (http://archive.ics.uci.edu/ml/machine-learning-databases/00267/data_banknote_authentication.txt).

- For N dimension and N class we use the Iris data set ("http://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data").

- For Binomial we use the spam base data set and replace all the non zero element in the data set with 1. To indicate the word had occurred in the email

- For Bernoulli we use the spam base data and replace all the non zero elements with number of times the words had occurred.

Result and discussion

One Dimension and two Class Dataset for Gaussian Discriminant Analysis

- Data set used for this was banknote authentication from UCI data repository.
- After applying the algorithm, the ten cross fold error came out to be around 15.5%
- The Confusion matrix formed is shown below with rows representing the classification result and the columns represent the true label for the two classes 0 and 1.

```
[[116  24]
 [ 16 118]]
```

- As shown in the matrix 116 0's where classified correct and 24 wrong as 1's.
- Also, 118 1's is classified correct where as 16 are classified wrong as 0's
- Giving us an accuracy of 85.4%.
- Other measures for each class is given below:

```
precision for class 0 is 0.828571
  Recall for class 0 is 0.878788
F measure for class 0 is 0.852941


precision for class 1 is 0.880597
  Recall for class 1 is 0.830986
F measure for class 1 is 0.855072
```

- We can see that we get high precision and recall for both the classes from the algorithm.

N Dimension and two Class Dataset for Gaussian Discriminant Analysis

- Data set used for this was banknote authentication from UCI data repository.
- Error with ten cross fold validation comes out of be around 1.6 %.
- The Confusion matrix formed is shown below with rows representing the classification result and the columns represent the true label for the two classes 0 and 1.

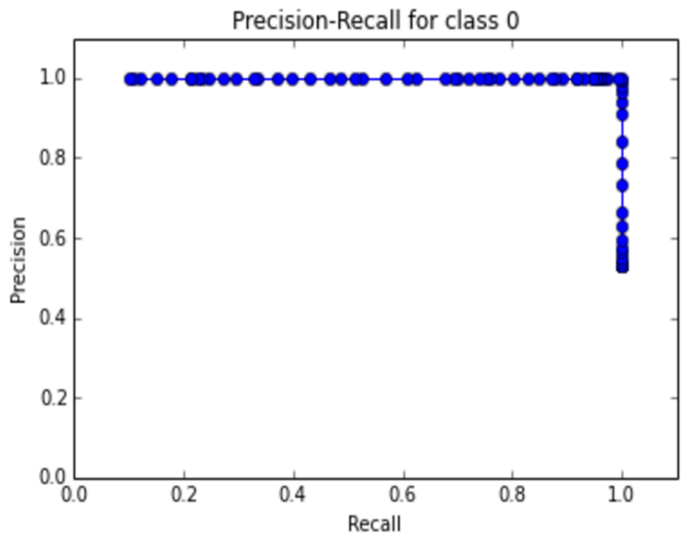
```
[[152   0]
 [  3 119]]
```

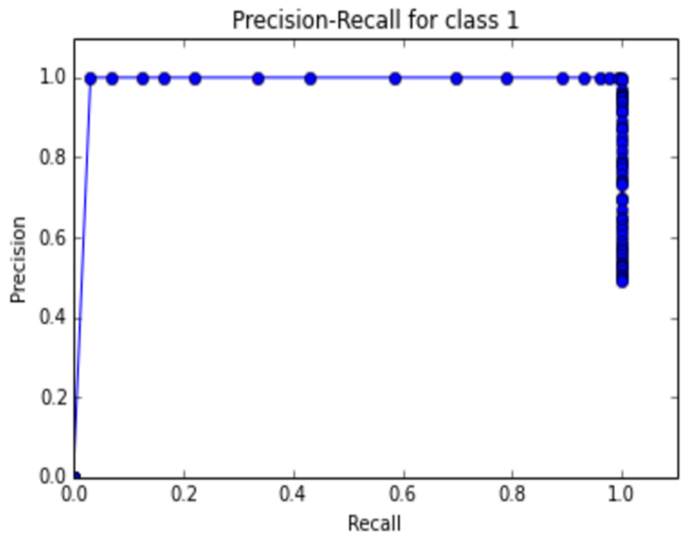
- As shown in the matrix all the 0's is classified correctly but 3 1's are classified as 0's.
- Accuracy given by the algorithm is around 98.9%
- Other measures for each class is given below:

```
precision for class 0 is 1.000000
  Recall for class 0 is 0.980645
F measure for class 0 is 0.990228


precision for class 1 is 0.975410
  Recall for class 1 is 1.000000
F measure for class 1 is 0.987552
```

- We see that precision is more for class 0 and recall is more for class 1. Also, F measure for class 0 is more.
- The reason for better precision is that there is no miss classification for class 0. So the value of True Positive and the sum of true positive and negative are same.
- The recall and precision curve as evident from the graph has value of 1. Suggesting that we have achieved the best possible value.





N Dimension and N Class Dataset for Gaussian Discriminant Analysis

- Data set used for this was Iris data set from UCI data repository.
- After applying the algorithm, the ten cross fold error came out to be around 3.33%
- The Confusion matrix formed is shown below with rows representing the classification result and the columns represent the true label for the three classes 0, 1 and 2.

```
[[ 7  0  0]
 [ 0 14  1]
 [ 0  1  7]]
```

- As shown in the matrix none of the 0's is classified wrong, but one 1's is classified as 2 and one 2's is classified as 1.

- Giving us an accuracy of 93.3%
- Other measures for each class is given below:

```
precision for class 0 is 1.000000
  Recall for class 0 is 1.000000
F measure for class 0 is 1.000000


precision for class 1 is 0.933333
  Recall for class 1 is 0.933333
F measure for class 1 is 0.933333


precision for class 2 is 0.875000
  Recall for class 2 is 0.875000
F measure for class 2 is 0.875000
```

- We get the highest result for class 0 as there is no misclassification.

Naïve Bayes with Bernoulli features

- Data set used for this is the spam base dataset from UCI data repository.
- All the non-zero values in columns 0-48 are converted to 1.
- After applying the algorithm, the ten cross fold error came out to be around 29.9%
- The Confusion matrix formed is shown below with rows representing the classification result and the columns represent the true label for the two classes 0 and 1.

```
[[460 161]
 [109 190]]
```

- As shown in the matrix 161 of the 0's is classified wrong, and 109 1's is classified wrong.
- Giving us an accuracy of 70.6%.
- Other measures for each class is given below:

```
precision for class 0 is 0.740741
  Recall for class 0 is 0.808436
F measure for class 0 is 0.773109


precision for class 1 is 0.635452
  Recall for class 1 is 0.541311
F measure for class 1 is 0.584615
```

- The percentage of misclassification for 1 is more than that of the class 0 evident by the the small F measure of class 1.
- Also, this is the reason for the recall and precision of class 1 being between 50 – 60 %.

Naïve Bayes with Binomial features

- Data set used for this is the spam base dataset from UCI data repository.
- All the non-zero values in columns 0-48 are converted to frequency of the word by multiplying it by length of each mail taken as 1000.
- After applying the algorithm, the ten cross fold error came out to be around 30.21%
- The Confusion matrix formed is shown below with rows representing the classification result and the columns represent the true label for the two classes 0 and 1.

```
[[286   9]
 [252 373]]
```

- As shown in the matrix 9 of the 0's is classified wrong, and 252 1's is classified wrong.
- Giving us an accuracy of 71.6%.
- Other measures for each class is given below:

```
precision for class 0 is 0.969492
 Recall for class 0 is 0.531599
F measure for class 0 is 0.686675


precision for class 1 is 0.596800
 Recall for class 1 is 0.976440
F measure for class 1 is 0.740814
```

- Changing the length of each mail from 1000 to 500 gave us better results.

```
Error by 10 cross fold:  0.286253418844

Confusion matrix

[[287  12]
 [274 347]]


ACCURACY:  0.689130434783


precision for class 0 is 0.959866
 Recall for class 0 is 0.511586
F measure for class 0 is 0.667442
```

```
precision for class 1 is 0.558776
 Recall for class 1 is 0.966574
F measure for class 1 is 0.708163
```

References

- Stackoverflow.com
- Pythonprogramming.net
- Coursera.com
- Element of statistical learning.

NAIVE BAYES BINOMIAL

5.

a. Model Selected:-

$$P(x_j^i \mid y=1) = \binom{p^i}{x_j^i} (\alpha_{j \mid y=1})^{x_j^i} (p^i - x_{j \mid y=1})^{(p^i - x_j^i)}$$

Computing the parameters:-

log likelihood can be written as

$$l(\theta) = \log \prod_{i=1}^{m} P(x^i \mid y^i, \theta) \cdot P(y^{(i)})$$

By Assumming all the ex$\overset{am}{\wedge}$ples are independent.

$$\Rightarrow \log \prod_{i=1}^{m} \left[\prod_{j=1}^{n} P(x_j^i \mid y^i, \theta) \right] \cdot P(y^{(i)})$$

Assumming that all features are also independent:

$$\Rightarrow \sum_{i=1}^{m} \sum_{j=1}^{n} \log P(x_j^i \mid y^i, \theta) + \sum_{j=1}^{m} \log P(y^{(i)})$$

$$\Rightarrow \sum_{i=1}^{m} \sum_{j=1}^{n} \log \left[\binom{p^i}{x_j^i} (\alpha_{j \mid y=y^{(i)}})^{x_j^i} (p^i - \alpha_{j \mid y=y^{(i)}})^{(p^i - x_j^i)} \right]$$

$$+ \sum_{i=1}^{m} \log P(y^{(i)}) \Rightarrow \text{Prior class probability.}$$

$\Rightarrow$ $\Theta^* = \underset{\Theta}{\arg\max}\ l(\Theta)$

we have the following parameter

$\Rightarrow$ $[\alpha_{1}/y=1 \quad \alpha_{2}/y=1 \quad \cdots \quad \alpha_{n}/y=1, \ \alpha_{1}/y=2 \quad \cdots \quad \alpha_{n}/y=2,$
$\alpha \ldots \quad \alpha_{n}/y=k]\ [\alpha_1 \ \cdots \ \alpha_k]$

$$\Rightarrow \frac{\partial l}{\partial \alpha_{i}|_{y=1}} = \frac{\partial}{\partial \alpha_{p}|_{y=1}} \left(\sum_{i=1}^{m} \log \binom{p^i}{x_p^i} + x_p^i \log(\alpha_p|y=1) + (p^i - x_p^i) \log(p^i - \alpha_p|y=1) \right)$$

$$+$$
$$0$$

$$\Rightarrow \sum_{i=1}^{m} \frac{x_p^i}{\alpha_p|y=1} + \sum_{i=1}^{m} \frac{(p^i - x_p^i)}{p^i - \alpha_p|y=1} = 0$$

let $x_p^i = a$ and $p^i - x_p^i = b$

$$\Rightarrow \therefore \alpha_p|y=1 = \frac{a}{a+b} = \frac{\sum_{i=1}^{m} \frac{x_p^i}{p^i}}{\sum_{i=1}^{m}}$$

$$\therefore \boxed{\alpha_{j/y=1} = \frac{\sum_{i=1}^{m} I(y^{(i)}=1)\, x_j^{(i)}}{\sum_{i=1}^{m} I(y^{(i)}=1)\, \#p^{(i)}}}$$

and α_i is the prior class ~~prior~~ probability of class i.