cs584 Assignment 2: Report

Arun Rajagopalan
Graduate student, A20360689
Department of Computer Science
Illinois Institute of Technology

March 20, 2016

Abstract:

The problem or the task is to implement techniques for generative learning. Datasets used are the Iris dataset available in UCI repository and the MobileMessageEnsemble dataset. It is crucial that we calculate and analyze the performance of our algorithms and we should use 10-fold cross validation for that purpose. It is also to be noted that we cannot use the direct python functions for learning and classification, but have to implement them ourselves.

1 Problem statement:

To implement techniques for generative learning. Two main parts of this assignment are Gaussian discriminant analysis and naïve Bayes classification with Bernoulli and Binomial features, where we have to implement functions for both. Also, the experiments need to conducted with subsets of taken datasets that have single feature and multiple features as well. In addition, outcomes achieved by varying the number of classes must also be recorded. Based on these experiment outcomes, we calculate the accuracy, precision, recall and F-Measure values, optionally plotting the precision-recall curve and finding area under it, to evaluate the performance of our classifier.

2 Proposed solution:

The solution intends to perform discriminant analysis and naïve Bayes classification using own functions and validate the results against in-built python functions. The major own functions implemented include those used for getting parameters of feature distribution, calculating membership (for both 1D and nD, 2 and k classes), predicting using a discriminant function which decides class based on membership values and to perform cross-validation. Precision-recall curve is drawn for 2 class nD gaussian discriminant analysis and naïve Bayes classification (both Bernoulli and Binomial).

3 Implementation details:

To implement the solution, python 2.7.3 was used. Main packages used include 'numpy' (for numerical processing and sklearn for some learning related functions. Direct python functions that do generative learning are not used, unless for validation.

Gaussian Discriminant Analysis:

This analysis is a generative model for classification where each class distribution is modeled as a Gaussian. The process involved is as follows. First, we read the input file and get attributes list and outcomes/labels list from them. For Iris (shuffled) dataset, which has characteristics of three categories of Iris flowers, the last column of the read file data is saved as outcomes and the remaining columns in the dataset are saved as attributes. With these attributes and outcomes lists, we are set to begin our experiments.

1D 2class:

For this case, we discard all attribute columns except any one at a time with only two classes. Then we split this input data into several folds and perform 10-fold cross validation with each fold being used exactly once for testing and 9 times for training.

Parameters: At each step of cross validation, we get mean and variance values of the attribute column for each class. These values are referred to as the parameters and they give us an idea about the feature distribution. We also calculate the prior class probability of each class which is nothing but the ratio of number of records of that class to the total number of records.

Membership: Now, for every record in the test set, we compute its membership in each class. This membership is calculated using the parameters with the formula given below.

Membership in a class = (-log(ClassVariance)-((0.5)*((x-ClassMean)**2/ClassVariance**2)) +log(ClassPrior))

The discriminant function will give the final prediction as that class where the record has higher membership.

Confusion Matrix: We can now calculate performance metrics, given that we have actual class and predicted class values. A confusion matrix is formed in this fashion: When the actual and prediction are same and if the predicted class value is 1 (Setosa), then it is a true positive (tPos) or else it is a true negative (tNeg). When the actual and prediction are not same and if the predicted class is 1, then it is taken as a false positive (fPos) or else it is a false negative (fNeg).

Metrics: From this matrix, we can calculate accuracy, precision, recall and f-measure values. Their formulae in python are as follows.

```
Accuracy = float(tPos+tNeg)/(tPos+tNeg+fNeg+fPos)
Precision = float(tPos)/(tPos+fPos) if tPos+fPos != 0 else 0.0
Recall = float(tPos)/(tPos+fNeg) if tPos+fNeg != 0 else 0.0
F-Measure = 2*float(Precision*Recall)/(Precision+Recall) if Precision+ Recall != 0 else 0.0
```

We also optionally plot the P-R curve or the precision recall curve where precision values are taken in Y axis and recall values are taken in X axis. For our experiments we use the precision_recall_curve function in sklearn to plot and auc function to get area under the curve.

For our ideal classifier, all the metrics above will be reasonably high, especially the F_Measure (since it is a single overall metric for the classifier) and the auc (Large area under curve means our classifier is good) values.

Verification: We also use sklearn's LinearDiscriminantAnalysis (Inbuilt) function to get predictions and compare it with our own functions for verification.

nD 2class and nD kclass:

The fundamental difference in this set of experiments when compared to the above is that here we select all the columns instead of just one. Thus our attribute list will have all columns, for all classes in case of k class and only 2 classes in case of 2 class. Outcomes list is same as the previous experiment for 2 class case and for the k class case, we take the entire original outcomes list. This is our input data.

Parameters: We perform the same cross validation here as well. In case of parameters, similar to the previous case, we gather mean of every column for every class and the prior class probability of that class. Instead of calculating the variance, here we calculate sigma value which is an array of covarinace matrices for each record in the input attributes list.

Membership: Calculating membership is also a bit different for the nD case. The formula is a follows.

Membership in a class = (-log(determinant(ClassSigma))- 0.5*(dot(dot(transpose(x-ClassMean), inverse(ClassSigma)), (x-ClassMean))) + log(ClassPrior))

With this membership value, just like the previous case, we tag that class value as prediction for a record where the record has the maximum membership.

Confusion matrix, Metrics and Verification: All these parts are done in a similar manner as the 1D case, except that for k-class our confusion matrix will have 9 values instead of just 4.

Naive Bayes classification:

The basic principle of naïve Bayes classification is that we assume the features to be independent as well. Although usually the features are inter-dependent, by making this assumption, we simplify our model and limit the parameters. This will be favorable in cases where there are lots of features, since we would still have plenty of information for training even though we give up the inter-record relations.

To perform experiments on Naive Bayes, we choose the MobileMessageEnsemble dataset, which is a synthetic ensemble of messages collected from user mobiles, annotated as spam or not spam (nospam). First of all, we read this datafile and save the first column as outcomes and the second column has attributes. Our input data is now set.

Bernoulli and Binomial cases:

In the Bernoulli case, we assume that our features have a Bernoulli distribution. For every term in every message (record) in the taken dataset, we check whether the term is present in our pre-built dictionary or not. If yes, we mark the term as 1 and if no, we mark the term as 0. Thus our features are effectively reduced from an array of strings (terms) to an array of 0s and 1s.

For Binomial, instead of just recording the presence or absence of a term, we count its relative frequency. This is achieved by getting the ratio of number of times the term occurs in a record to the total number of terms in the record.

Parameters: Here, alpha is our parameter which is the distribution of a term in the dataset. Each and every term/record in the dataset will have an alpha value. The parameter is calculated as below.

Parameter alpha = (spamTotal+sConst)/(spamTermCount+2*sConst),(nospamTotal+sConst)/(nospamTermCount+2*sConst)

spamTotal and nospamTotal are the total number of records where a term appears and is marked as spam or nospam respectively. In case of Bernoulli, spamTermCount and nospamTermCount are just number of records that have been marked spam or nospam. In case of Binomial, it is the total number of terms that are marked as spam or nospam. If a word is not present in the records (in a given class), it will disturb the spam or no spam probability and hence we add sConst (Laplace smoothing) to ensure that missing terms don't affect our estimate. Thus we calculate the parameter alpha value (which is spam and nospam frequency) for each term in the training data and add it to our dictionary.

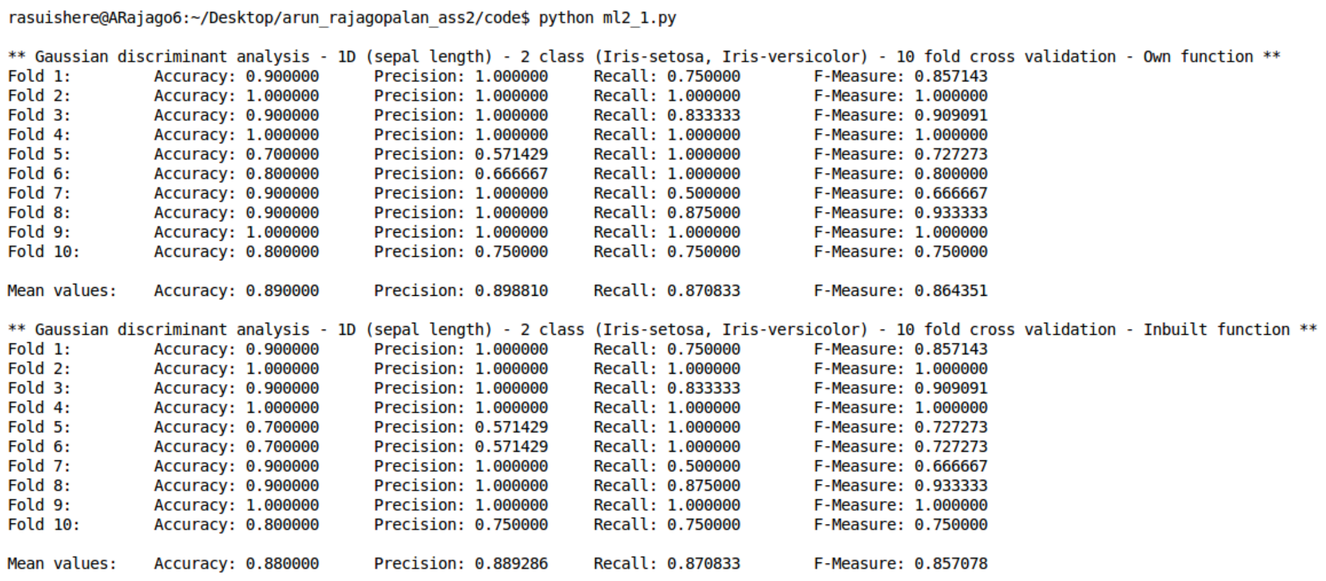
Prediction: For every record in test data, we split out the terms and add up the log of spam/nospam frequency of each term to get spamMeasure and nospamMeasure for each record. We multiply these measures with class priors and assign the record that class where it has higher measure.

Confusion matrix, Metrics and Verification: These calculations are done exactly how they are done in GDA 1D case.

4 Results and discussion:

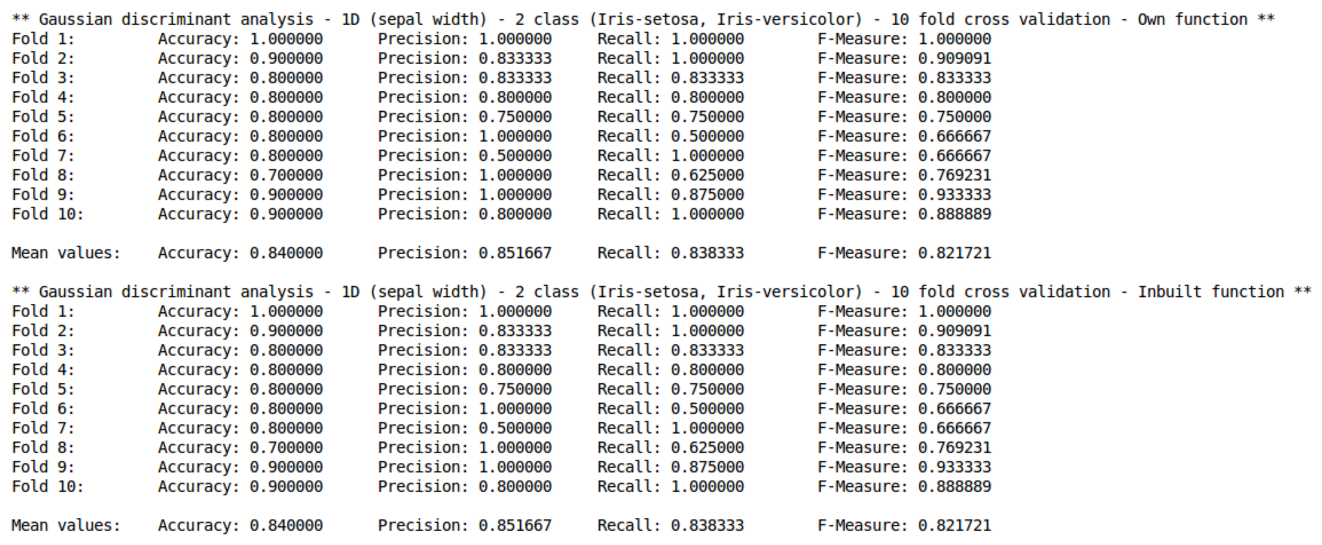
GDA 1D 2 class case outputs:

- As explained before, in this case, we select only one attribute to perform classification. We get very good metrics when we choose petal length or petal width as our attribute (F-Measure of 1) and relatively good results when we choose sepal length or sepal width. (F-Measure of 0.85 and 0.82)
- From the outputs, it is evident that petal length and petal width have distinct values for each class but sepal length and width values are not very distinct.
- From the P-R curve, we get good area under curve meaning that our classifier is performing good.
- The outputs achieved from own function and inbuilt function are mostly the same.



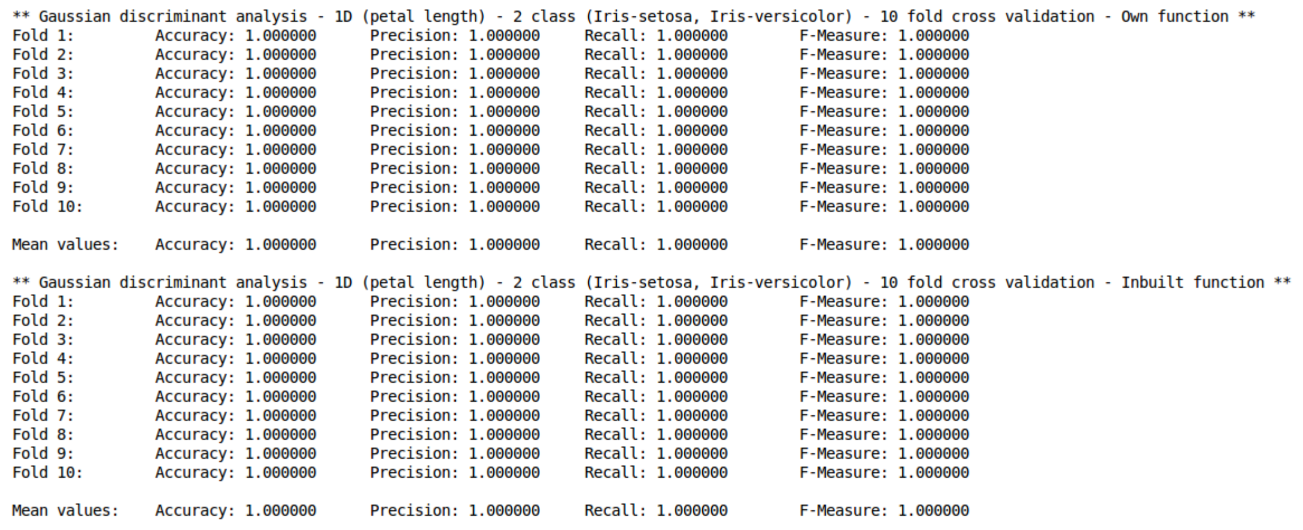
```
rasuishere@ARajago6:~/Desktop/arun_rajagopalan_ass2/code$ python ml2_1.py

** Gaussian discriminant analysis - 1D (sepal length) - 2 class (Iris-setosa, Iris-versicolor) - 10 fold cross validation - Own function **
Fold 1:         Accuracy: 0.900000      Precision: 1.000000     Recall: 0.750000        F-Measure: 0.857143
Fold 2:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 3:         Accuracy: 0.900000      Precision: 1.000000     Recall: 0.833333        F-Measure: 0.909091
Fold 4:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 5:         Accuracy: 0.700000      Precision: 0.571429     Recall: 1.000000        F-Measure: 0.727273
Fold 6:         Accuracy: 0.800000      Precision: 0.666667     Recall: 1.000000        F-Measure: 0.800000
Fold 7:         Accuracy: 0.900000      Precision: 1.000000     Recall: 0.500000        F-Measure: 0.666667
Fold 8:         Accuracy: 0.900000      Precision: 1.000000     Recall: 0.875000        F-Measure: 0.933333
Fold 9:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 10:        Accuracy: 0.800000      Precision: 0.750000     Recall: 0.750000        F-Measure: 0.750000

Mean values:    Accuracy: 0.890000      Precision: 0.898810     Recall: 0.870833        F-Measure: 0.864351

** Gaussian discriminant analysis - 1D (sepal length) - 2 class (Iris-setosa, Iris-versicolor) - 10 fold cross validation - Inbuilt function **
Fold 1:         Accuracy: 0.900000      Precision: 1.000000     Recall: 0.750000        F-Measure: 0.857143
Fold 2:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 3:         Accuracy: 0.900000      Precision: 1.000000     Recall: 0.833333        F-Measure: 0.909091
Fold 4:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 5:         Accuracy: 0.700000      Precision: 0.571429     Recall: 1.000000        F-Measure: 0.727273
Fold 6:         Accuracy: 0.700000      Precision: 0.571429     Recall: 1.000000        F-Measure: 0.727273
Fold 7:         Accuracy: 0.900000      Precision: 1.000000     Recall: 0.500000        F-Measure: 0.666667
Fold 8:         Accuracy: 0.900000      Precision: 1.000000     Recall: 0.875000        F-Measure: 0.933333
Fold 9:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 10:        Accuracy: 0.800000      Precision: 0.750000     Recall: 0.750000        F-Measure: 0.750000

Mean values:    Accuracy: 0.880000      Precision: 0.889286     Recall: 0.870833        F-Measure: 0.857078
```

Illustration 1: GDA 1D Sepal Length - Own and Inbuilt function



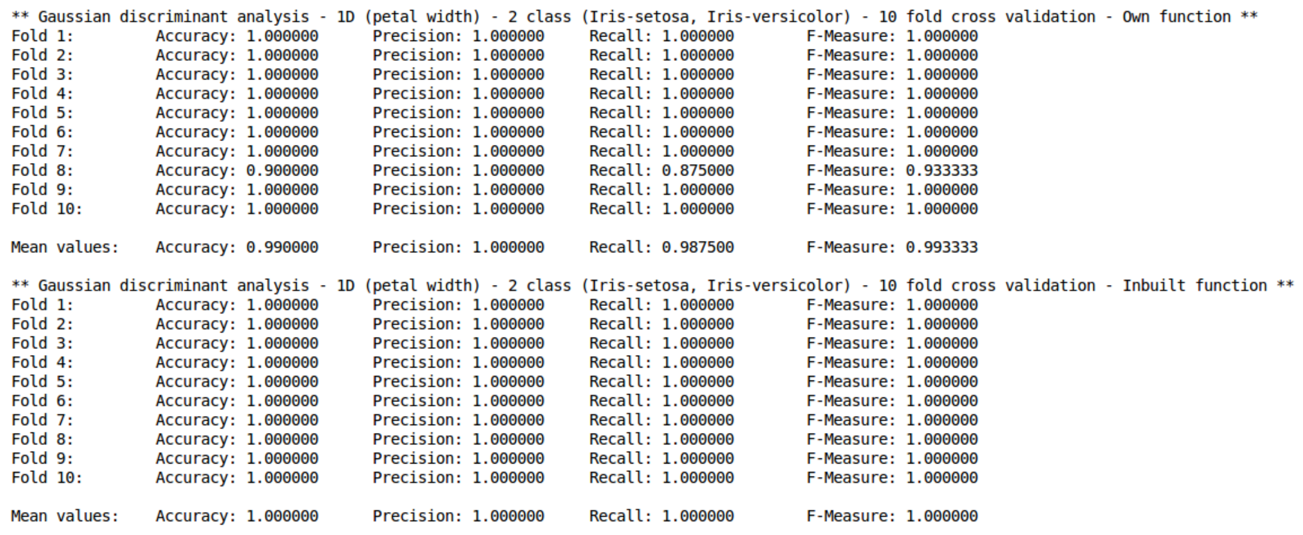
```
** Gaussian discriminant analysis - 1D (sepal width) - 2 class (Iris-setosa, Iris-versicolor) - 10 fold cross validation - Own function **
Fold 1:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 2:         Accuracy: 0.900000      Precision: 0.833333     Recall: 1.000000        F-Measure: 0.909091
Fold 3:         Accuracy: 0.800000      Precision: 0.833333     Recall: 0.833333        F-Measure: 0.833333
Fold 4:         Accuracy: 0.800000      Precision: 0.800000     Recall: 0.800000        F-Measure: 0.800000
Fold 5:         Accuracy: 0.800000      Precision: 0.750000     Recall: 0.750000        F-Measure: 0.750000
Fold 6:         Accuracy: 0.800000      Precision: 1.000000     Recall: 0.500000        F-Measure: 0.666667
Fold 7:         Accuracy: 0.800000      Precision: 0.500000     Recall: 1.000000        F-Measure: 0.666667
Fold 8:         Accuracy: 0.700000      Precision: 1.000000     Recall: 0.625000        F-Measure: 0.769231
Fold 9:         Accuracy: 0.900000      Precision: 1.000000     Recall: 0.875000        F-Measure: 0.933333
Fold 10:        Accuracy: 0.900000      Precision: 0.800000     Recall: 1.000000        F-Measure: 0.888889

Mean values:    Accuracy: 0.840000      Precision: 0.851667     Recall: 0.838333        F-Measure: 0.821721

** Gaussian discriminant analysis - 1D (sepal width) - 2 class (Iris-setosa, Iris-versicolor) - 10 fold cross validation - Inbuilt function **
Fold 1:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 2:         Accuracy: 0.900000      Precision: 0.833333     Recall: 1.000000        F-Measure: 0.909091
Fold 3:         Accuracy: 0.800000      Precision: 0.833333     Recall: 0.833333        F-Measure: 0.833333
Fold 4:         Accuracy: 0.800000      Precision: 0.800000     Recall: 0.800000        F-Measure: 0.800000
Fold 5:         Accuracy: 0.800000      Precision: 0.750000     Recall: 0.750000        F-Measure: 0.750000
Fold 6:         Accuracy: 0.800000      Precision: 1.000000     Recall: 0.500000        F-Measure: 0.666667
Fold 7:         Accuracy: 0.800000      Precision: 0.500000     Recall: 1.000000        F-Measure: 0.666667
Fold 8:         Accuracy: 0.700000      Precision: 1.000000     Recall: 0.625000        F-Measure: 0.769231
Fold 9:         Accuracy: 0.900000      Precision: 1.000000     Recall: 0.875000        F-Measure: 0.933333
Fold 10:        Accuracy: 0.900000      Precision: 0.800000     Recall: 1.000000        F-Measure: 0.888889

Mean values:    Accuracy: 0.840000      Precision: 0.851667     Recall: 0.838333        F-Measure: 0.821721
```

Illustration 2: GDA 1D Sepal Width - Own and Inbuilt function



```
** Gaussian discriminant analysis - 1D (petal length) - 2 class (Iris-setosa, Iris-versicolor) - 10 fold cross validation - Own function **
Fold 1:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 2:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 3:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 4:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 5:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 6:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 7:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 8:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 9:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 10:        Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000

Mean values:    Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000

** Gaussian discriminant analysis - 1D (petal length) - 2 class (Iris-setosa, Iris-versicolor) - 10 fold cross validation - Inbuilt function **
Fold 1:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 2:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 3:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 4:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 5:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 6:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 7:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 8:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 9:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 10:        Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000

Mean values:    Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
```

Illustration 3: GDA 1D Petal Length - Own and Inbuilt function



```
** Gaussian discriminant analysis - 1D (petal width) - 2 class (Iris-setosa, Iris-versicolor) - 10 fold cross validation - Own function **
Fold 1:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 2:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 3:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 4:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 5:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 6:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 7:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 8:         Accuracy: 0.900000      Precision: 1.000000     Recall: 0.875000        F-Measure: 0.933333
Fold 9:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 10:        Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000

Mean values:    Accuracy: 0.990000      Precision: 1.000000     Recall: 0.987500        F-Measure: 0.993333

** Gaussian discriminant analysis - 1D (petal width) - 2 class (Iris-setosa, Iris-versicolor) - 10 fold cross validation - Inbuilt function **
Fold 1:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 2:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 3:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 4:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 5:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 6:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 7:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 8:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 9:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 10:        Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000

Mean values:    Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
```

Illustration 4: GDA 1D Petal Width - Own and Inbuilt function

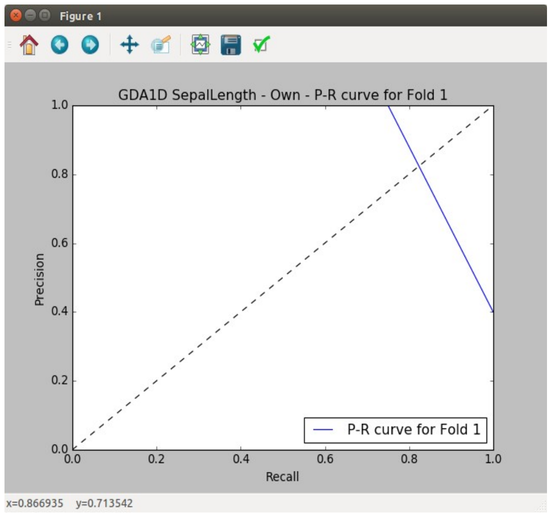


Illustration 5: GDA 1D P-R Curve Sepal Length Own function

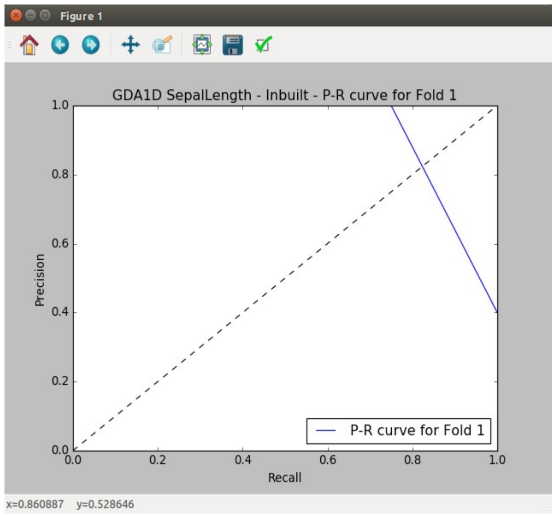
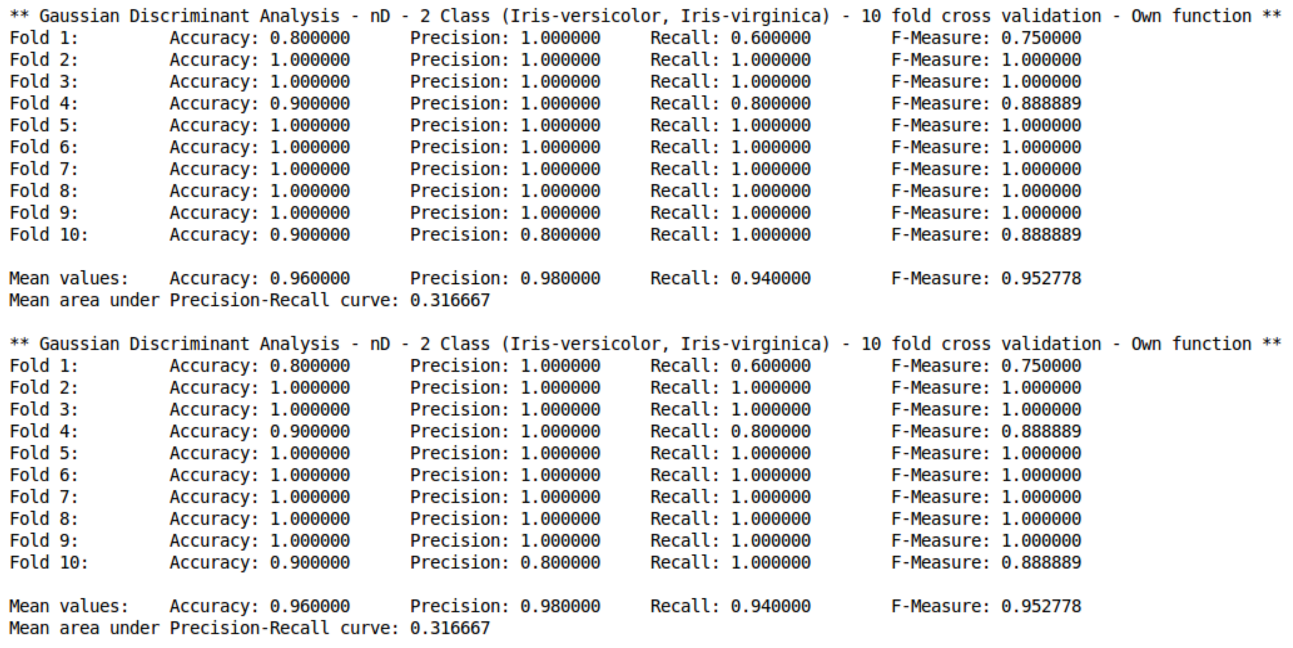


Illustration 6: Illustration 5: GDA 1D P-R Curve Sepal Length Inbuilt function

GDA nD 2 class case outputs:

- Although the best metrics that we get in this case are not as high as the best of 1D case, we get more consistent or generalized results here.
- From the P-R curve, we get decent area under curve and all our metrics are still near 1 meaning that our classifier is performing good.
- Here too, the outputs achieved from own function and inbuilt function are exactly the same.



```
** Gaussian Discriminant Analysis - nD - 2 Class (Iris-versicolor, Iris-virginica) - 10 fold cross validation - Own function **
Fold 1:         Accuracy: 0.800000      Precision: 1.000000     Recall: 0.600000        F-Measure: 0.750000
Fold 2:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 3:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 4:         Accuracy: 0.900000      Precision: 1.000000     Recall: 0.800000        F-Measure: 0.888889
Fold 5:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 6:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 7:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 8:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 9:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 10:        Accuracy: 0.900000      Precision: 0.800000     Recall: 1.000000        F-Measure: 0.888889

Mean values:    Accuracy: 0.960000      Precision: 0.980000     Recall: 0.940000        F-Measure: 0.952778
Mean area under Precision-Recall curve: 0.316667

** Gaussian Discriminant Analysis - nD - 2 Class (Iris-versicolor, Iris-virginica) - 10 fold cross validation - Own function **
Fold 1:         Accuracy: 0.800000      Precision: 1.000000     Recall: 0.600000        F-Measure: 0.750000
Fold 2:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 3:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 4:         Accuracy: 0.900000      Precision: 1.000000     Recall: 0.800000        F-Measure: 0.888889
Fold 5:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 6:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 7:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 8:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 9:         Accuracy: 1.000000      Precision: 1.000000     Recall: 1.000000        F-Measure: 1.000000
Fold 10:        Accuracy: 0.900000      Precision: 0.800000     Recall: 1.000000        F-Measure: 0.888889

Mean values:    Accuracy: 0.960000      Precision: 0.980000     Recall: 0.940000        F-Measure: 0.952778
Mean area under Precision-Recall curve: 0.316667
```

Illustration 7: GDA nD 2 class (Versicolor, Virginica) All features – Own and Inbuilt functions

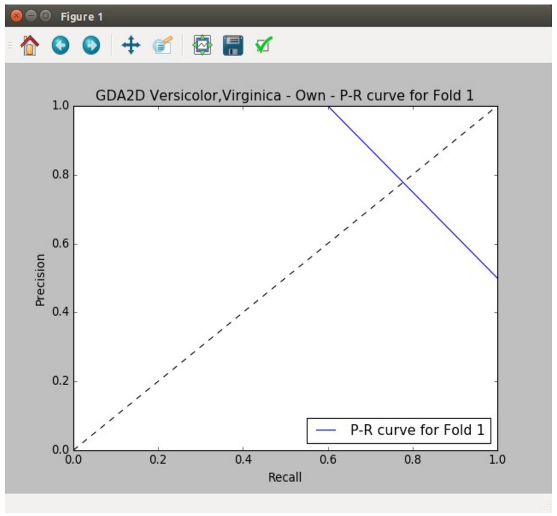


Illustration 9: GDA nD P-R Curve Versicolor,Virginica Own function

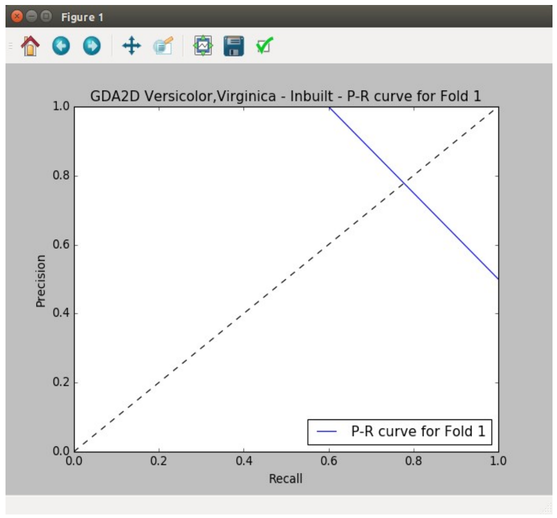
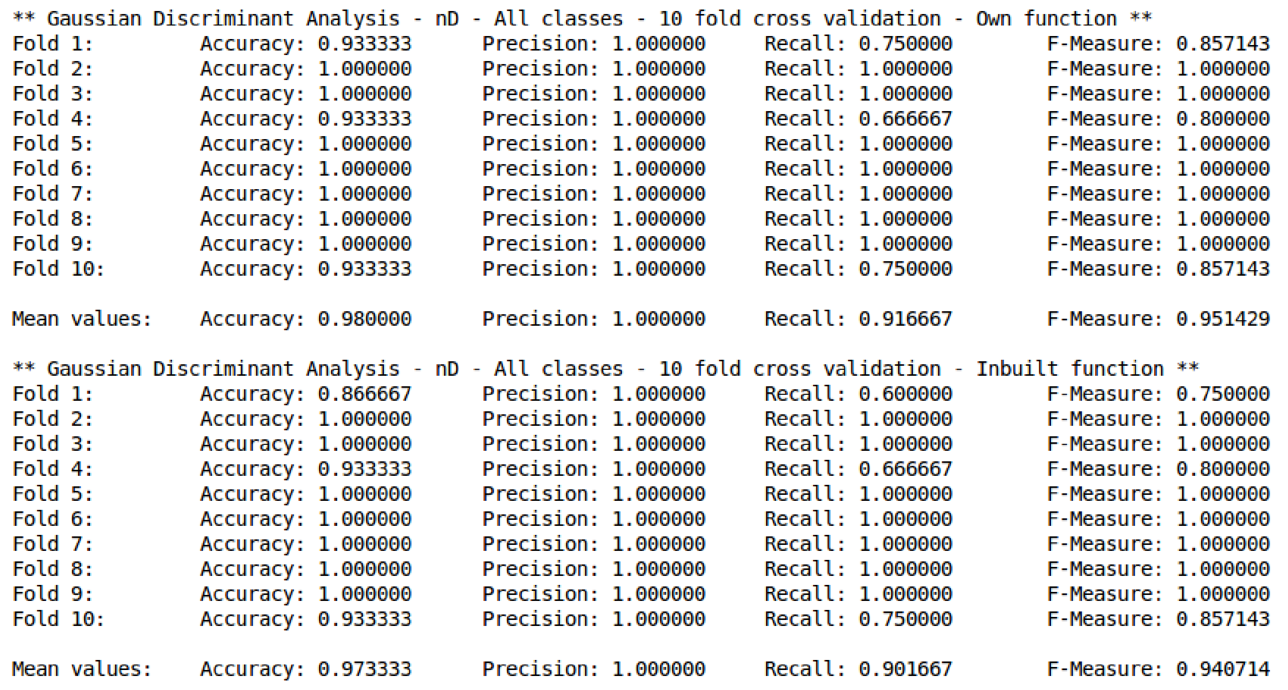


Illustration 8: GDA nD P-R Curve Versicolor,Virginica Inbuilt function

GDA nD k class case outputs:

- The metrics that we get here are comparable with the 2 class case. Our metrics are all near one which is an indication that our classifier is still performing good.
- The outputs achieved from own function and inbuilt function are mostly the same for this case as well.



```
** Gaussian Discriminant Analysis - nD - All classes - 10 fold cross validation - Own function **
Fold 1:         Accuracy: 0.933333      Precision: 1.000000      Recall: 0.750000        F-Measure: 0.857143
Fold 2:         Accuracy: 1.000000      Precision: 1.000000      Recall: 1.000000        F-Measure: 1.000000
Fold 3:         Accuracy: 1.000000      Precision: 1.000000      Recall: 1.000000        F-Measure: 1.000000
Fold 4:         Accuracy: 0.933333      Precision: 1.000000      Recall: 0.666667        F-Measure: 0.800000
Fold 5:         Accuracy: 1.000000      Precision: 1.000000      Recall: 1.000000        F-Measure: 1.000000
Fold 6:         Accuracy: 1.000000      Precision: 1.000000      Recall: 1.000000        F-Measure: 1.000000
Fold 7:         Accuracy: 1.000000      Precision: 1.000000      Recall: 1.000000        F-Measure: 1.000000
Fold 8:         Accuracy: 1.000000      Precision: 1.000000      Recall: 1.000000        F-Measure: 1.000000
Fold 9:         Accuracy: 1.000000      Precision: 1.000000      Recall: 1.000000        F-Measure: 1.000000
Fold 10:        Accuracy: 0.933333      Precision: 1.000000      Recall: 0.750000        F-Measure: 0.857143

Mean values:    Accuracy: 0.980000      Precision: 1.000000      Recall: 0.916667        F-Measure: 0.951429

** Gaussian Discriminant Analysis - nD - All classes - 10 fold cross validation - Inbuilt function **
Fold 1:         Accuracy: 0.866667      Precision: 1.000000      Recall: 0.600000        F-Measure: 0.750000
Fold 2:         Accuracy: 1.000000      Precision: 1.000000      Recall: 1.000000        F-Measure: 1.000000
Fold 3:         Accuracy: 1.000000      Precision: 1.000000      Recall: 1.000000        F-Measure: 1.000000
Fold 4:         Accuracy: 0.933333      Precision: 1.000000      Recall: 0.666667        F-Measure: 0.800000
Fold 5:         Accuracy: 1.000000      Precision: 1.000000      Recall: 1.000000        F-Measure: 1.000000
Fold 6:         Accuracy: 1.000000      Precision: 1.000000      Recall: 1.000000        F-Measure: 1.000000
Fold 7:         Accuracy: 1.000000      Precision: 1.000000      Recall: 1.000000        F-Measure: 1.000000
Fold 8:         Accuracy: 1.000000      Precision: 1.000000      Recall: 1.000000        F-Measure: 1.000000
Fold 9:         Accuracy: 1.000000      Precision: 1.000000      Recall: 1.000000        F-Measure: 1.000000
Fold 10:        Accuracy: 0.933333      Precision: 1.000000      Recall: 0.750000        F-Measure: 0.857143

Mean values:    Accuracy: 0.973333      Precision: 1.000000      Recall: 0.901667        F-Measure: 0.940714
```

Illustration 10: GDA nD All class All features – Own and Inbuilt functions

Naive Bayes with Bernoulli and Binomial features:

- In this set of experiments, we find that the metrics and area under curve are much better for

Binomial case. Since Bernoulli takes only the presence or absence of a term into account its performance is relatively poor when compared with Binomial which uses the relative frequency of terms.

- F-Measure for Bernoulli is around 0.5 which is very average, but that for Binomial stands strong at 0.81. Even accuracy and precision are better for Naive Bayes with Binomial features

```
** Naive Bayes with Bernoulli features - 10 fold cross validation - Own Function **
Fold 1:         Accuracy: 0.708075      Precision: 0.361991     Recall: 1.000000        F-Measure: 0.531561
Fold 2:         Accuracy: 0.668737      Precision: 0.259259     Recall: 1.000000        F-Measure: 0.411765
Fold 3:         Accuracy: 0.720497      Precision: 0.311224     Recall: 1.000000        F-Measure: 0.474708
Fold 4:         Accuracy: 0.710145      Precision: 0.299492     Recall: 0.967213        F-Measure: 0.457364
Fold 5:         Accuracy: 0.678423      Precision: 0.276995     Recall: 0.983333        F-Measure: 0.432234
Fold 6:         Accuracy: 0.738589      Precision: 0.329787     Recall: 1.000000        F-Measure: 0.496000
Fold 7:         Accuracy: 0.705394      Precision: 0.323671     Recall: 0.971014        F-Measure: 0.485507
Fold 8:         Accuracy: 0.678423      Precision: 0.298643     Recall: 1.000000        F-Measure: 0.459930
Fold 9:         Accuracy: 0.670124      Precision: 0.293333     Recall: 1.000000        F-Measure: 0.453608
Fold 10:        Accuracy: 0.684647      Precision: 0.276190     Recall: 1.000000        F-Measure: 0.432836

Mean values:    Accuracy: 0.696306      Precision: 0.303059     Recall: 0.992156        F-Measure: 0.463551
Area under Precision-Recall curve: 0.395690

** Naive Bayes with Binomial features - 10 fold cross validation - Own Function **
Fold 1:         Accuracy: 0.952381      Precision: 0.787879     Recall: 0.975000        F-Measure: 0.871508
Fold 2:         Accuracy: 0.925466      Precision: 0.613636     Recall: 0.964286        F-Measure: 0.750000
Fold 3:         Accuracy: 0.956522      Precision: 0.756410     Recall: 0.967213        F-Measure: 0.848921
Fold 4:         Accuracy: 0.946170      Precision: 0.721519     Recall: 0.934426        F-Measure: 0.814286
Fold 5:         Accuracy: 0.954357      Precision: 0.750000     Recall: 0.950000        F-Measure: 0.838235
Fold 6:         Accuracy: 0.952282      Precision: 0.746835     Recall: 0.951613        F-Measure: 0.836879
Fold 7:         Accuracy: 0.931535      Precision: 0.687500     Recall: 0.956522        F-Measure: 0.800000
Fold 8:         Accuracy: 0.943983      Precision: 0.724138     Recall: 0.954545        F-Measure: 0.823529
Fold 9:         Accuracy: 0.935685      Precision: 0.692308     Recall: 0.954545        F-Measure: 0.802548
Fold 10:        Accuracy: 0.941909      Precision: 0.678571     Recall: 0.982759        F-Measure: 0.802817

Mean values:    Accuracy: 0.944029      Precision: 0.715880     Recall: 0.959091        F-Measure: 0.818872
Area under Precision-Recall curve: 0.707718
```

Illustration 11: Naive Bayes with Bernoulli and Binomial features – Own function

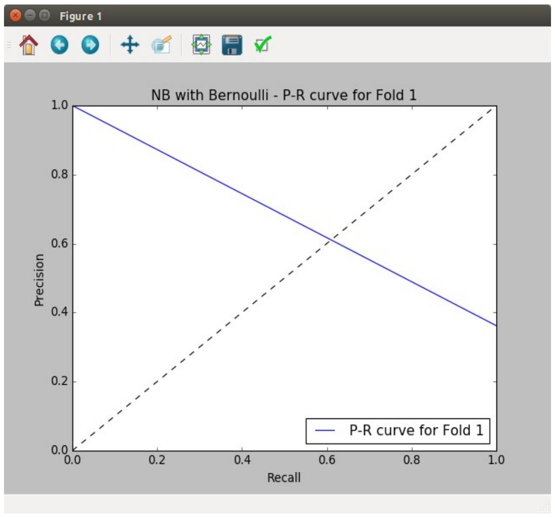


Illustration 12: P-R Curve NB with Bernoulli features - Own function

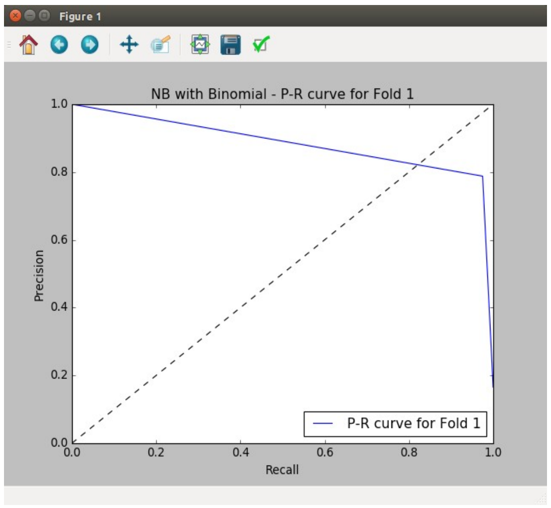


Illustration 13: P-R Curve NB with Binomial features - Own function

5 References: http://openclassroom.stanford.edu/MainFolder/CoursePage.php?course=MachineLearning