**CS697A – Topic in Computer Science – Machine Learning**

Assignment 2

**Submitted by: Prachi Panse**

**NetID: ve3568**

**Q1:** Parametric Classification: Using each of the 64 input features separately as the single input dimension, use parametric classification, assuming that the input is distributed according to a Gaussian. Report the training and test errors for the case of each of the 64 features. Which feature(s) give the best test performance?

***Answer:***

Column #1 to #64 are features and column #65 is the class code, so considering values till column #64 for calculations.

Step 1: Calculate mean and variance for training data, for class 6 and class 9 separately.

Step 2: Calculate discriminant function for training data. For each cell in 200 \* 64 matrix calculate discriminant function. g6(x) and g9(x) where x is each cell in 200 \* 64 matrix.

Step 3: Predicting misclassification in the training data: compare g6(x) and g9(x) for each cell. Consider the greater value class and label that cell with this greater valued class. Continue this process for every cell in the training data.

Step 4: Counting the errors per column: If any cell which is labeled with class 9 but it was originally in the class 6, is misclassified by class 6 and hence increment the error count of class 6 for that column by 1. Also applies vice a versa to class 9. This step gives us the column wise errors in each class.

Step 5: Total training error: for each column and each class, d(theta)

Error = Total misclassified / number of rows

Step 6: Followed the same process on the testing data from step 2.

Step 7: To identify the best feature from the testing data, skip the features for which variance is zero for any class. For the remaining features, add both classes’ error feature wise and a column with minimum error value is the best feature among all.

For the given training and testing data:

training\_final\_error: [0.0, 0.0, 0.315, 0.325, 0.345, 0.32, 0.0, 0.0, 0.0, 0.31, 0.295, 0.39, 0.295, 0.085, 0.0, 0.0, 0.0, 0.275, 0.45, 0.385, 0.19, 0.0, 0.0, 0.0, 0.0, 0.47, 0.395, 0.37, 0.16, 0.025, 0.1, 0.0, 0.0, 0.26, 0.1, 0.26, 0.295, 0.37, 0.365, 0.0, 0.0, 0.285, 0.005, 0.1, 0.45, 0.445, 0.27, 0.0, 0.0, 0.495, 0.225, 0.2, 0.37, 0.535, 0.175, 0.4, 0.0, 0.465, 0.31, 0.265, 0.36, 0.26, 0.255, 0.44]

testing\_final\_error: [0.0, 0.0, 0.22, 0.38, 0.42, 0.30, 0.0, 0.0, 0.0, 0.26, 0.22, 0.23, 0.35, 0.07, 0.0, 0.0, 0.0, 0.30, 0.51, 0.36, 0.15, 0.0, 0.0, 0.0, 0.0, 0.54, 0.34, 0.33, 0.16, 0.08, 0.13, 0.0, 0.0, 0.19, 0.08, 0.20, 0.25, 0.45, 0.37, 0.0, 0.0, 0.23, 0.01, 0.09, 0.38, 0.51, 0.34, 0.0, 0.0, 0.45, 0.28, 0.17, 0.46, 0.48, 0.27, 0.45, 0.0, 0.46, 0.23, 0.33, 0.41, 0.31, 0.37, 0.49]

best\_features: ['3 : 0.22160664819944598', '4 : 0.3767313019390582', '5 : 0.42105263157894735', '6 : 0.30193905817174516', '10 : 0.25761772853185594', '11 : 0.22437673130193905', '12 : 0.23545706371191136', '13 : 0.3490304709141274', '14 : 0.07202216066481995', '18 : 0.29916897506925205', '19 : 0.5152354570637119', '20 : 0.3573407202216066', '21 : 0.14958448753462603', '26 : 0.5457063711911357', '27 : 0.34349030470914127', '28 : 0.3296398891966759', '29 : 0.15789473684210525', '30 : 0.08033240997229917', '31 : 0.1329639889196676', '34 : 0.19390581717451524', '35 : 0.08310249307479224', '36 : 0.20498614958448755', '37 : 0.24930747922437674', '38 : 0.4515235457063712', '39 : 0.37119113573407203', '42 : 0.23268698060941828', **'43 : 0.0110803324099723**', '44 : 0.09141274238227147', '45 : 0.3767313019390582', '46 : 0.5124653739612188', '47 : 0.34349030470914127', '50 : 0.4515235457063712', '51 : 0.27977839335180055', '52 : 0.16620498614958448', '53 : 0.4598337950138504', '54 : 0.47645429362880887', '55 : 0.2659279778393352', '56 : 0.4515235457063712', '58 : 0.4598337950138504', '59 : 0.2299168975069252', '60 : 0.33240997229916897', '61 : 0.407202216066482', '62 : 0.31024930747922436', '63 : 0.3684210526315789', '64 : 0.4930747922437673']

There are 45 features in training dataset with non-zero value and 45 features in testing dataset with non-zero value.

Among these all values in testing dataset, for identifying the best feature, minimum value is for feature number 43rd (feature index staring from 1), which is 0.011. Therefore, this is the best feature among all.

**Q2:** Use all the 64 features, assume that inputs are 64 dimensional Gaussians, and assume that for each class the covariance matrix is different. Report the training and test confusion matrices and errors. **Hint:** eliminate features that have covariance zero.

***Answer:***

Step 1: Calculate feature wise mean vector for training data, for class 6 and class 9 separately.

Step 2: Calculate covariance matrix for class 6 and class 9 using the relevant rows in training data.

Step 3: Calculate discriminant functions g6(x) and g9(x) for each row from the training data, there will be 200 scalar values for g6(x) and 200 scalar values for g9(x), use separately calculated covariance matrices and mean vectors for each class to calculate discriminant functions.

Step 4: Compare the value for g6(x) and g9(x) for each row and label it with the class with greater value.

Step 5: Error counting: check for the label assigned in the previous step, if the original class label is same as assigned class label then no error if there is a mismatch between the original class and assigned class then count an error for the original class.

Step 6: Accordingly calculate total errors= total errors for class 6 + total errors for class 9.

Step 7: Next, process on the testing data, there is no need to recalculate the feature wise mean vector and covariance matrix, directly calculate discriminant function for each row same as we did in training data.

Step 8: follow step #4 and #5 to calculate total errors in testing data.

For the given training and testing data:

total\_training\_errors: 0

training\_errors6: 0

training\_errors9: 0

total\_testing\_errors: 18

testing\_errors6: 16

testing\_errors9: 2

* Confusion Matrix for training dataset:

|  |  |  |
| --- | --- | --- |
|  | **Predicted Class** | |
| **True Class** | **Yes** | **No** |
| **Yes** | True Positive  100 | False Negative  0 |
| **No** | False Positive  0 | True Negative  100 |

Error rate = (FN + FP) / N = 0

* Confusion Matrix for testing dataset:

|  |  |  |
| --- | --- | --- |
|  | **Predicted Class** | |
| **True Class** | **Yes** | **No** |
| **Yes** | True Positive  165 | False Negative  16 |
| **No** | False Positive  2 | True Negative  178 |

Error rate = (FN + FP) / N = (16 + 2) /361 = 0.0498

**Q3:** Repeat Q2, assuming that all the class covariance matrices are the same.

***Answer:***

Step 1: Calculated feature wise mean vector for training data, for class 6 and class 9 separately.

Step 2: Calculated covariance matrix for the entire training data, there was no need to calculate different covariance matrices for each class as they have common covariance matrix.

Step 3: Calculated discriminant functions g6(x) and g9(x) for each row from the training data, using separate mean vector for each class but common covariance matrix, there will be 200 scalar values for g6(x) and 200 scalar values for g9(x).

Step 4: Compared the value for g6(x) and g9(x) for each row and labelled it with the class with greater value.

Step 5: Error counting: checked for the label assigned in the previous step, if the original class label is same as assigned class label then no error if there is a mismatch between the original class and assigned class then counted an error for the original class.

Step 6: Accordingly, calculated “total errors= total errors for class 6 + total errors for class 9”.

Step 7: Next, processed on the testing data, there was no need to recalculate the mean and covariance matrix, directly calculated discriminant function for each row same as we did in training data.

Step 8: Followed step #4 and #5 to calculate total errors in testing data.

For the given training and testing data:

total\_training\_errors: 0

training\_errors6: 0

training\_errors9: 0

total\_testing\_errors: 0

testing\_errors6: 0

testing\_errors9: 0

* Confusion Matrix for training dataset:

|  |  |  |
| --- | --- | --- |
|  | **Predicted Class** | |
| **True Class** | **Yes** | **No** |
| **Yes** | True Positive  100 | False Negative  0 |
| **No** | False Positive  0 | True Negative  100 |

Error rate = (FN + FP) / N = 0

* Confusion Matrix for testing dataset:

|  |  |  |
| --- | --- | --- |
|  | **Predicted Class** | |
| **True Class** | **Yes** | **No** |
| **Yes** | True Positive  181 | False Negative  0 |
| **No** | False Positive  0 | True Negative  180 |

Error rate = (FN + FP) / N = 0

**Q4:** Use the first 10 features in Q1 that gave the best test performance and repeat Q2. Compare the test error performance you got to Q2.

***Answer:***

Step 1: Considered 10 best features from testing data which are giving minimum error in question 1. Removed all features except these 10 from training as well as testing data.

Step 2: Calculated feature wise mean vector for training data, for class 6 and class 9 separately.

Step 2: Calculated covariance matrix separately for class 6 and class 9 using the relevant rows in training data.

Step 3: Calculated discriminant functions g6(x) and g9(x) for each row from the training data, used separately calculated covariance matrices and mean vectors for each, there will be 200 scalar values for g6(x) and 200 scalar values for g9(x).

Step 4: Compared the value for g6(x) and g9(x) for each row and then labelled it with the class with greater value.

Step 5: Error counting: checked for the label assigned in the previous step, if the original class label was same as assigned class label then no error if there was a mismatch between the original class and assigned class then counted an error for the original class.

Step 6: Accordingly, calculated “total errors= total errors for class 6 + total errors for class 9”.

Step 7: Next, processed on the testing data, there was no need to recalculate the feature wise mean vector and covariance matrix, directly calculated discriminant function for each row same as we did in training data.

Step 8: followed step #4 and #5 to calculate total errors in testing data.

For the given training and testing data:

total\_training\_errors: 17

training\_error6: 17

training\_error9: 0

total\_testing\_errors: 62

testing\_errors6: 62

testing\_errors9: 0

* Confusion Matrix for training dataset:

|  |  |  |
| --- | --- | --- |
|  | **Predicted Class** | |
| **True Class** | **Yes** | **No** |
| **Yes** | True Positive  83 | False Negative  17 |
| **No** | False Positive  0 | True Negative  100 |

Error rate = (FN + FP) / N = (17+0) / 200 = 0.085

* Confusion Matrix for testing dataset:

|  |  |  |
| --- | --- | --- |
|  | **Predicted Class** | |
| **True Class** | **Yes** | **No** |
| **Yes** | True Positive  119 | False Negative  62 |
| **No** | False Positive  0 | True Negative  180 |

Error rate = (FN + FP) / N = (62+0) / 361 = 0.171