ÖZYEĞİN UNIVERSITY

Introduction to Machine Learning and Artificial Neural Networks

CS 554

HOMEWORK 1 Report
Parametric Classification

By: *Amin Deldari Alamdari*
S033174

Course Given By:

Prof. Ethem ALPAYDIN

Faculty of Engineering

Computer Science Deparment

1. INTRODUCTION

Parametric Classification in machine learning is a technique that necessitates an understanding of the statistical characteristics of the classification task. It involves calculating the likelihood of each class at any point in the pattern space and then selecting the most probable class for an unknown pattern based on these probabilities. This estimation is usually done using a training set, but the exact statistics can never be known precisely due to the restrictions of the available data. Consequently, parametric classification can only achieve sub-optimal classifications in real-world scenarios.

Parametric Classification is a machine learning approach that utilizes a fixed set of parameters to summarize data, regardless of the quantity of training examples.This approach involves two steps:

- selecting a form for the function, and
- learning coefficients for the function from the training data.

This algorithm is based on a mathematical model that establishes the connection between inputs and outputs, making it more restrictive than non-parametric algorithms, but also quicker and easier to train. Parametric algorithms are most suitable for issues where the input data is well-defined and foreseeable. Examples of parametric algorithms include linear regression, logistic regression, and neural networks.

The advantages of parametric machine learning algorithms includes they are easier to comprehend and interpret results, quick to learn from data, does not require a lot of training data and can still perform well even if the data fit is not ideal, can handle large datasets and can accommodate a wide range of functional forms, and can yield better prediction models. However, they have some disadvantages including limited complexity, needs a large amount of training data to estimate the mapping function, and they takes more time to train as it usually has more parameters to train.

2. PROCEDURE AND METHODS

In this homework, we are given two different **.csv** files as datasets, one for *training* our approach and the other for *testing* it. The goal is to use examples from the training dataset to calculate the class priors and parameters of the *Gaussian* densities, as specified in the homework instructions. We are required to:

1. Creating a graph that displays the likelihood and posterior distributions, as well as the training and testing dataset examples, on the same plot.

2. Determine the effectiveness of our model by computing the 0-1 loss and constructing 3-by-3 confusion matrices for both the training and testing datasets.

2.1. LIKELIHOOD AND POSTERIOR DISTRIBUTIONS

We must import the *matplotlib* library into our environment, as follows:

2.1.1. Loading Data

We begin by loading our dataset into the workspace. To do this, we must include the following code snippet at the start of our code:

We must sort the data in our training set into three classes. To do this, we use the following approach:

The Figure 2.1 visualizes the instances inside the *training dataset* based on their classes.

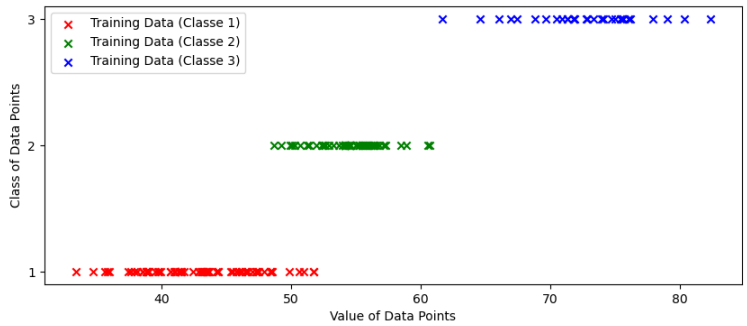

Figure 2.1: Training Dataset Visualization.

2.1.2. Calculating Class Priors

To find the class priors of each class we need to use the following equation:

$$\text{Prior Probability of a Class} = \frac{\text{Number of Instances of the Class}}{\text{Total Number of Instances}} \tag{2.1}$$

The *Prior Probability of Classes* are given in Table 2.1.

Class	Prior Probability
Class 1	0.4667
Class 2	0.3333
Class 3	0.2

Table 2.1: Prior Probabilities of Classes

2.1.3. Calculating Means and Variances

We can calculate the mean of a class by taking the sum of the values of each class and dividing it by the total number of examples in that class. Equation 2.2 can be used for this purpose.

$$\text{Mean} = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{2.2}$$

where n is the number of data points in the class dataset and x_i represents value of each individual data point. For calculating the variance of a class of dataset we can use the Equation 2.3 as follows:

$$\text{Variance} = \frac{1}{n}\sum_{i=1}^{n} (x_i - \bar{x})^2 \tag{2.3}$$

where n is the number of data points in the class dataset, x_i represents value of each individual data point and $\bar{x}$ is the mean of the dataset for each class where we found in previous section.

The values in Table 2.2 demonstrate the *mean* and *variance* of each class that were calculated.

Class	Mean Value	Variance
Class 1	42.9903	18.4885
Class 2	54.4919	6.8559
Class 3	72.6917	20.4053

Table 2.2: Mean and Variance Values of Classes

2.1.4. Calculating and plotting likelihood and posterior distributions

As stated on the Homework description, class densities are assumed to be *Gaussian*. Therefore, the likelihood of a dataset x given parameters μ and σ^2 in a Gaussian distribution is calculated using the probability density function (PDF):

$$f(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{2.4}$$

where x is the value for which we want to find the likelihood, μ is the mean of the distribution, and σ^2 is the variance of the distribution.

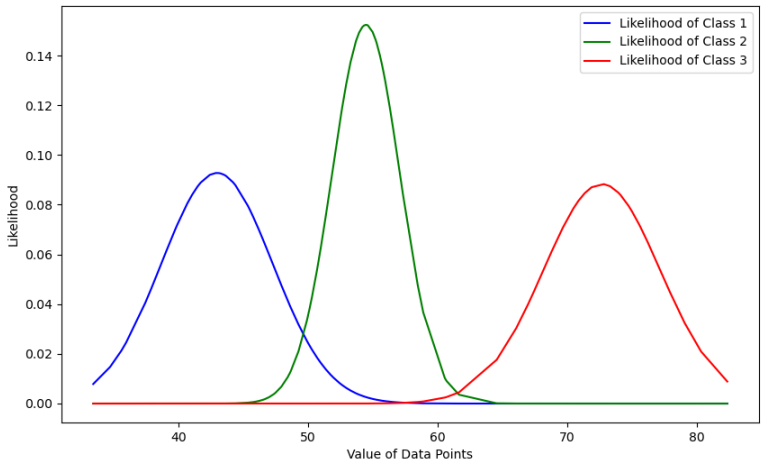

Figure 2.2: Likelihood Visualization of Classes in Training Dataset.

To calculate the posterior probability of a class given an input data point using Bayes' theorem, you can use the formula:

$$P(\text{Class}|\text{Data}) = \frac{P(\text{Data}|\text{Class}) \times P(\text{Class})}{P(\text{Data})} \tag{2.5}$$

where $P(\text{Class}|\text{Data})$ is the posterior probability of the class given the data, $P(\text{Data}|\text{Class})$ is the likelihood of the data given the class, $P(\text{Class})$ is the prior probability of the class, and $P(\text{Data})$ is the probability of the data (evidence).

In the case of Gaussian distributions with known means and variances and assuming equal priors,

the formula for the posterior probability of a class given an input value x is simplified to:

$$\text{Posterior}(\text{Class}) = \text{Likelihood}(x|\text{Class}) \times \text{Prior}(\text{Class}) \tag{2.6}$$

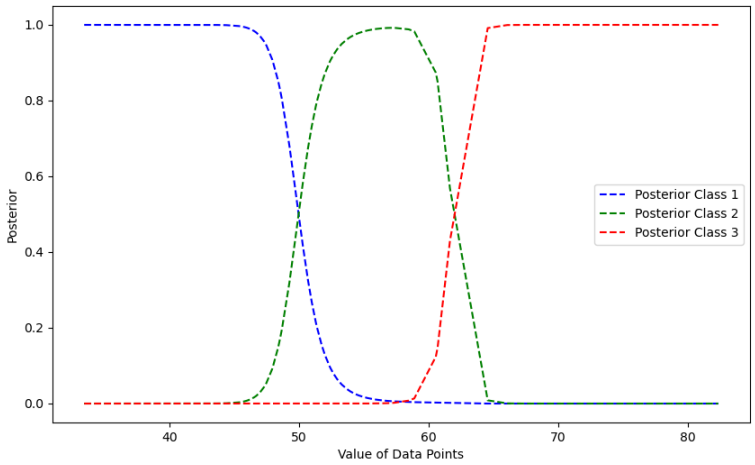

Figure 2.3: Posterior Distribution Visualization of Classes in Training Dataset.

Based on the first requirement of the homework we need to plot the likelihood and posterior distributions, together with the training and test data instances on the same plot by using different colors/symbols for different distributions and classes.

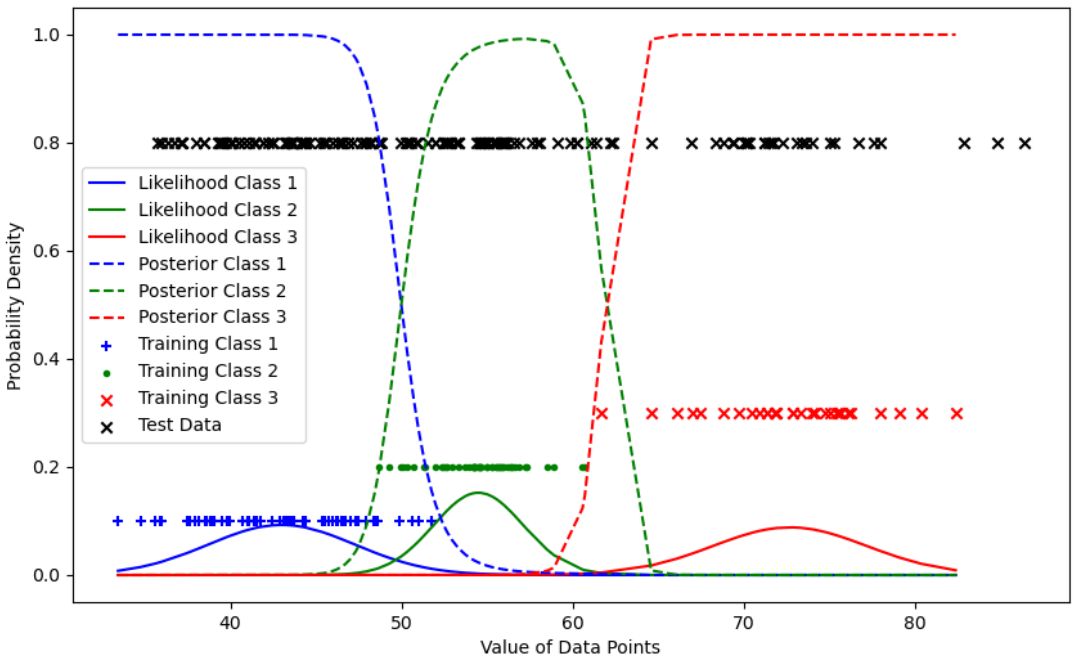

Figure 2.4: Likelihood and Posterior Distributions Visualization of Classes in Training Dataset and Data Points in Training and Test Datasets.

2.2. PERFORMANCE OF THE MODEL

In the second part of the homework, we required to calculate the performance of our model by assuming 0-1 loss and using *3-by-3 confusion matrices*. The effectiveness of a parametric classification model can be assessed by computing various metrics such as accuracy, precision, recall, and F1 score. A confusion matrix is a helpful tool for this purpose. It gives a breakdown of the model's predictions into four categories: true positives, false positives, true negatives, and false negatives. By counting the number of true positives, false positives, true negatives, and false negatives, one can calculate the accuracy, precision, recall, and F1 score of the model.

2.2.1. Confusion Matrix

A confusion matrix is a table that is used to evaluate the performance of a classification model. It displays the number of correct and incorrect predictions, divided by class. It is composed of four elements: True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN). These values can be used to calculate various metrics.

2.2.2. Steps to Calculate Performance using Confusion Matrix

Step 1: Generate Predictions

Given the model's predictions and the true class labels, generate the predictions.

Step 2: Construct the Confusion Matrix

Create a matrix that contains the counts of True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN) for each class.

Step 3: Calculate Metrics

From the confusion matrix, we can calculate various performance metrics:

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \tag{2.7}$$

$$\text{Precision} = \frac{TP}{TP+FP} \tag{2.8}$$

$$\text{Recall} = \frac{TP}{TP+FN} \tag{2.9}$$

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2.10}$$

Step 4: Interpretation

We need to interpret the metrics obtained to understand the model's performance.

2.3. CALCULATING THE PERFORMANCE OF THE MODEL

Confusion Matrix for Training Data:

$$\begin{bmatrix} 66 & 4 & 0 \\ 3 & 47 & 0 \\ 0 & 1 & 29 \end{bmatrix}$$

Confusion Matrix for Test Data:

$$\begin{bmatrix} 67 & 3 & 0 \\ 2 & 47 & 1 \\ 0 & 0 & 30 \end{bmatrix}$$

3. RESULTS AND DISCUSSION

From the results obtained through parametric classification using Gaussian distributions and the confusion matrices calculated for both the training and testing data, several inferences and conclusions can be drawn:

- **Model Performance:** Confusion matrices provide insight into how well a model is classifying the data. The diagonal elements of the confusion matrix indicate the number of correct classifications for each class, while the off-diagonal elements show misclassifications.
- **Accuracy:** Precision measures the ratio of correctly predicted positive instances to the total predicted positive instances. Recall measures the ratio of correctly predicted positive instances to the total actual positive instances. F1-score is the harmonic mean of precision and recall. The confusion matrix can be used to calculate metrics such as accuracy, precision, recall, and F1-score. Accuracy is the proportion of correctly classified instances out of the total number of instances. Precision is the ratio of correctly identified positive cases to the total number of positive cases predicted. Recall is the ratio of correctly identified positive cases to the total number of actual positive cases. F1-score is the harmonic mean of precision and recall.
- **Class-Specific Performance:** The confusion matrix can be used to determine which classes are often mistaken for one another. For instance, it may demonstrate that Class 1 is more likely to be misidentified as Class 2 than Class 3.
- **Overfitting or Underfitting:** The difference between the performance of a model on the training and testing data can be an indication of either overfitting or underfitting. If the model performs much better on the training data than on the unseen test data, it could be overfit. On the other hand, if the performance is consistent between the training and test data, it could be a sign of a well-generalized model.
- **Optimization and Further Analysis:** If the model's performance is not up to par, it can be improved by making adjustments. This could include feature engineering, refining the model's parameters, or using different classification algorithms to increase its effectiveness.

- **Assumptions Validation:** This approach supposes that each class has a Gaussian distribution. However, the data may not be an exact match for these assumptions, which can lead to a decrease in the model's effectiveness. Examining the model's weaknesses can help to identify areas where the assumptions may not be accurate.
- **Decision Boundaries:** Parametric classification enables the formation of decision boundaries based on estimated distributions for each class. Gaining insight into how the model distinguishes between classes could be facilitated by visualizing these boundaries.
- **Predictive Power:** In order to assess the model's predictive power, it is necessary to evaluate its capacity to accurately generalize to data that has not been seen before. The confusion matrix of the test data is a key factor in this assessment.

3.1. INTERPRETING THE TRAINING DATASET CONFUSION MATRIX

The confusion matrix is a 3×3 table that gives an overview of the performance of a parametric classification model on a training dataset with three different categories. It provides insight into how well the model is doing. In our case the confusion matrix of training instances:

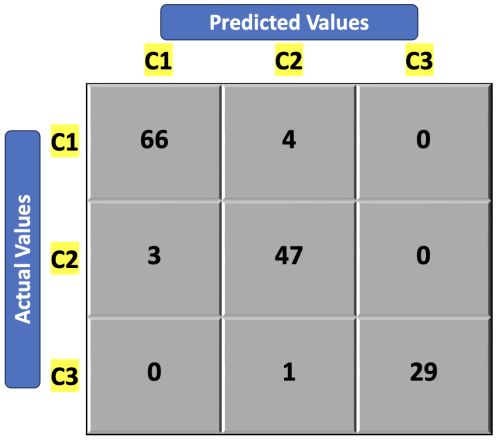

Figure 3.1: Confusion Matrix for Training Dataset Instances.

Matrix Elements and Interpretation: Each row in the matrix is associated with an actual class, and each column is indicative of the model's predicted class.

Accuracy: Most of the accurate forecasts are located on the main diagonal, which shows the model's accuracy.

Correct Predictions: Examining the diagonal elements:

- Class 1 (top-left) shows 66 correct predictions out of 70 instances.

- Class 2 (middle) demonstrates 47 correct predictions out of 50 instances.
- Class 3 (bottom-right) reveals 29 correct predictions out of 30 instances.

Misclassifications: The elements that are not on the diagonal of the matrix indicate misclassifications, such as 4 cases from Class 1 being incorrectly classified as Class 2 and 3 cases from Class 2 being incorrectly classified as Class 1.

Imbalance or Bias: It appears that the model's predictions between Class 1 and Class 2 have a slight imbalance or bias, as indicated by the non-zero values in the off-diagonal elements.

Sensitivity and Specificity: This matrix allows us to calculate the sensitivity (true positive rate) and specificity (true negative rate) for each class. Sensitivity is the proportion of actual positive cases that are correctly identified, while specificity is the proportion of actual negative cases that are correctly identified.

Overall Model Assessment: The confusion matrix helps us to comprehend how well the model fits the data, but it is essential to evaluate the model on data that it has not seen before in order to determine its ability to generalize and how it will perform on new data.

To sum up, the training data confusion matrix provides an overview of the model's performance across different classes, showing where mistakes are made and which classes are more likely to be misclassified, which can be used to make potential changes to the model or data preprocessing.

3.2. INTERPRETING THE TESTING DATASET CONFUSION MATRIX

In our case the confusion matrix of testing instances:

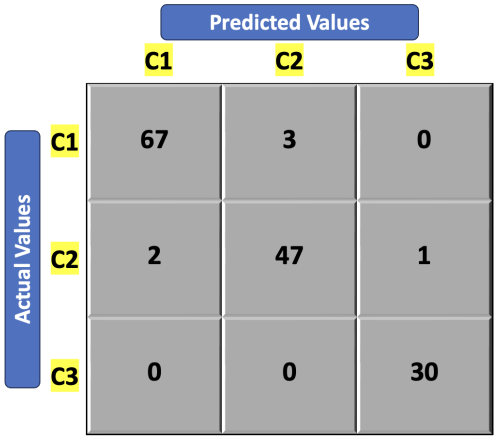

Figure 3.2: Confusion Matrix for Testing Dataset Instances.

This matrix represents the counts for three classes. From this matrix, the following inferences can be made:

- For Class 1:
  - 67 instances were correctly classified as Class 1.
  - 3 instances of Class 1 were misclassified as Class 2.
  - 0 instances of Class 1 were misclassified as Class 3.
- For Class 2:
  - 47 instances were correctly classified as Class 2.
  - 2 instances of Class 2 were misclassified as Class 1.
  - 1 instance of Class 2 was misclassified as Class 3.
- For Class 3:
  - 30 instances were correctly classified as Class 3.
  - 0 instances of Class 3 were misclassified as Class 1 or Class 2.

Observations and Conclusions:

- The model shows relatively good performance, especially in differentiating Class 3.
- The primary misclassifications occur between Class 1 and Class 2.
- Metrics like accuracy, precision, recall, and F1-score are recommended for a more comprehensive evaluation of the model's performance.

It's crucial to test the model on a separate dataset to assess its generalization ability.