Group 31

Aditi Verma (S20180010006)
Nethra Gunti (S20180010061)
Kavya Nemmoju (S20180010078)
Dasari Jayasree (S20180010047)

ML Assignment 01

September 11, 2020

Abstract

The aim of this project is to implement the Bayes Classifier and the Naive Bayes Classifier for the given dataset. The dataset in use is an IRIS dataset divided into a training set (75% of total samples) and test set (remaining 25% of the samples). For implementing the Bayes Classifier, we have used the Multivariate Normal Distribution of the training data to train the model, while assuming that the priors are equal. Whereas, for implementing the Naive Bayes Classifier, we have used the N-fold Cross Validation method with N=5.

EDA

The IRIS dataset provided to us has been split into two parts: the test file (containing 75% of the samples) and the test file (containing 25% of the samples). The complete dataset consists of 50 samples of 3 classes: Iris Setosa, Iris Versicolour, Iris Virginica. The dataset also provides the following features for each sample, which can be used to predict the class labels: sepal length in cm, sepal width in cm, petal length in cm, petal width in cm and ground values of class labels.

Train Data:

The train file contains 112 samples in total. The distribution of the samples is as follows:

1. Iris-setosa: 41 samples
2. Iris-versicolor: 32 samples
3. Iris-virginica: 39 samples.

Test Data:

The test file contains 38 samples in total. The distribution of the samples is as follows:

4. Iris-setosa: 9 samples
5. Iris-versicolor: 18 samples
6. Iris-virginica: 11 samples.

Libraries

This project has been done using Python 3.8. The following are the libraries used in the project:

1. **Numpy** version 1.19.1

    About:

    NumPy is a python library that allows the usage of multi-dimensional arrays and matrices along with high level mathematical functions and operations.

2. **Pandas** version 1.1.1

    About:

    Pandas is a python library that is majorly used for Data Manipulation and Analysis. It offers data structures and operations for manipulating numerical tables and time series.

Approach Summary

In accordance with the assignment requirements, the model of choice for class prediction was the Bayes classifier. In this codebase, we aim to implement two variations of the Bayes classifier: one which assumes Multivariate Gaussian distribution of the feature values and uses their joint probabilities to make class predictions; as well as a Naive Bayes variation which assumes strong (naive) independence between the features along with a Gaussian distribution. Both the models have been implemented in Python3 and perform differently on the given dataset.

Procedure

1. **Bayes Classifier**

   Bayes Classifier is a simple probabilistic classifier which works on the Bayes Theorem. We used the training set to train the model and then predicted results for the test set. The detailed procedure is as follows:

   a. **Reading the files:** Read the test and training sets into respective DataFrames using Pandas
   b. **Preprocessing:** Separate the training set based on the “class” values.
   c. **Fitting the model:** Calculate the Sample Mean and Sample Covariance with the help of array/matrix operations that NumPy offers.
   d. **Calculating the Probability:** With the given information that the priors are equal (1/3), the probability can be calculated assuming the dataset to be a Multivariate Normal Distribution.

   $$P(x) \ = \ 1/3 \ \times \ [exp(-1/2 \ \times \ (x-\mu)\,^{t}\Sigma^{-1} \ \ (x-\mu))] \ \div \ [2\Pi^{d/2} \, |\Sigma^{1/2}|]$$

   e. **Predicting the Class:** The class for a test sample can now be predicted using the above probability. Calculate the probabilities of a sample belonging to each class and then choose the class with the highest probability.
   f. **Defining accuracy:** The accuracy for this classifier can be calculated by comparing the predicted values and the actual values.

   $$correctly\ predicted\ values \ \times \ 100 \ \div \ actual\ values$$

2. **Naive Bayes Classifier**

   Naive Bayes classifier is a simple probabilistic classifier based on applying Bayes' theorem with strong (naïve) independence assumptions between the features. This is one of the simplest Bayesian network models. The detailed procedure for the building the model is as follows:

a. Reading the dataset files: The files were read using the pandas library and made into their respective DataFrames.
b. Preprocessing: The train was separated into different DataFrames based on the "class" values.
c. Fitting the model: The model was fitted using closed-form expressions, making use of linear statistical functions such as mean and standard deviation of the features for samples in each class. Numpy mathematical functions were used for the same.
d. Calculating the posterior probabilities: The posterior probabilities of classes for each sample were calculated using the Gaussian distribution based on the statistics obtained in the previous step for each feature and aggregating them. The probability function is:

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

e. Predicting the class: The class for a test sample can now be predicted using the above probability. The sample is assigned to the class with the highest posterior class probability.

Results

The results we got for the above models were fairly positive.

1. Bayes Classifier: The multivariate classifier has a
    a. 5-fold cross validation accuracy: 95.45%
    b. Test set accuracy: 100.0%
2. Naive Bayes Classifier: This Bayes classifier has a
    a. 5-fold cross validation accuracy: 93.64%
    b. Test set accuracy: 94.74%