DEPARTMENT OF COMPUTER & SOFTWARE ENGINEERING

COLLEGE OF E&ME, NUST, RAWALPINDI

EC-350 Artificial Intelligence and Decision Support System

LAB MANUAL – 11

Course Instructor: Assoc Prof Dr Arsalan Shaukat

Lab Engineer: Kashaf Raheem

Student Name: ______________________________

Degree/ Syndicate: ______________________________

LAB # 11: UNIVARIATE/ MULTIVARIATE BAYESIAN CLASSIFICATION MODEL

Lab Objective:

- To implement univariate or multivariate Bayesian classification model in Python

Hardware/Software required:

Hardware: Desktop/ Notebook Computer
Software Tool: Python 3.10.0

Lab Description:

Bayesian is one of the simplest probabilistic classifiers that classifies a candidate test vector based on Bayes Rule:

$$P(w_i|\, x) = \frac{P(x|\, w_i) * P(w_i)}{P(x)}$$

where $P(w_i|\, x)$ is the posterior probability telling the conditional probability of the class w_i against the given feature vector x, $P(x|\, w_i)$is the likelihood probability of the feature vector x given the class w_i, $P(w_i)$ is the prior probability of class w_i and $P(x)$ is the evidence of feature vector x. Bayesian classifier can be used to classify both numerical and categorical test vectors. For classifying numerical test vectors, Bayesian classifier assumes that the likelihood probabilities of the underlying test vector (whether univariate or multivariate) originate from Gaussian probability density function. For the univariate test vector x, the Gaussian distribution can be computed through:

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

where μ is the mean and σ is the standard deviation of the univariate Gaussian distribution computed using following formulas:

$$\mu = \sum_{k=1}^{n} \frac{x_k}{n}$$

$$\sigma^2 = \frac{\sum_{K=1}^{n} (x_k - \mu)^2}{n}$$

Multivariate Gaussian distribution can be computed as:

$$P(x;\ \mu, \Sigma) = \frac{1}{(2\pi)^{d/2}} |\Sigma|^{-1/2} e^{\left(-\frac{1}{2}(x-\mu)\Sigma^{-1}(x-\mu)^T\right)}$$

where μ represents a mean vector, d represents dimension and Σ represents a dxd covariance matrix.

$$\Sigma = \frac{\sum_{k=1}^{n} (x_k - \mu)(x_k - \mu)^t}{n}$$

The algorithm to implement Bayesian classification model is given below:

Algorithm: Bayesian Classifier

```
Input:  Training data X
        Training data labels Y
        Sample x to classify
Output: Decision $y_p$ about sample x
Compute prior probabilities ‘p1’ for the first-class
Compute prior probabilities ‘p2’ for the second-class
Compute likelihood for the test sample x against first-class training samples through univariate or multivariate Gaussian Distribution
Compute likelihood for the test sample x against second-class training samples through univariate or multivariate Gaussian Distribution
Compute posterior probabilities ‘P1’ and ‘P2’
Compute evidence by summing P1 and P2.
Normalize P1 by dividing it with the evidence
Normalize P2 by dividing it with the evidence
If P1 >= P2
    $y_p = 0$
else
    $y_p = 1$
return $y_p$
```

Lab Tasks:

Task1

Use the given diabetes dataset and classify it using multivariate Bayesian classification model:

a) First create a Python script and load ‘Diabeties.csv’ file.
b) Identify features and classes from the loaded dataset.
c) Perform 2-fold cross validation on the dataset by splitting it into testing and training parts.
d) Implement a Bayesian classifier using the above algorithm and use training dataset to classify each of the sample within testing dataset.
e) Compute the accuracy from the predicted test samples.

Steps for task 1:

1. Load the csv file and split the dataset into training and testing by 2-fold-cross validation.
2. Separate the training dataset w.r.t its class and save them in different matrix, you can use numpy. unique(label_col) to identify unique labels. Compute the prior probabilities for all the classes.
3. The columns in the given dataset represents a unique feature such as “Age”. find the mean for all features for both classes and save them in an array.
4. Now calculate covariance matrix all both labels. You can either write the code for mathematical expression given above or can use buit-In command numpy.cov(feature_matrix).
5. Finding determinant using built-in function numpy.linalg.det(covariance_matix)
6. For test sample X find multivariate gaussian density i.e., likelihood for each class using their respective means and covariance matrixes. You can use scipy.stats.multivariate_normal.pdf(x,mean,covarainace)
7. Compute posterior probabilities ‘P1’ and ‘P2’ and classify.

Task2

Write a Python script to implement univariate Bayesian classification model for the following dataset and classify the given test sample:

2	0
1	0

3	0
1.5	0
4.1	0
6	1
5	1
7	1
8	1
7.6	1
5	?