**National University of Computer and Emerging Sciences**



**Lab Manual 07**

**CL461-Artificial Intelligence Lab**

|  |  |
| --- | --- |
| Course Instructor | Eesha Tur Razia Babar |
| Lab Instructor (s) | Abdul Rehman  Mateen Fatima |
| Section | BDS-6A |
| Semester | Spring 2024 |

**Lab Task:**

You have been given a Google Colab starter code and a dataset, you must perform the following tasks on the given dataset:

**Coding Exercise 1: Data Exploration and Preprocessing**

**Give Ans**

1. Data Quality

* Common data quality problems

What common data problems can be observed in the dataset

(Answer in 5 Lines)

* Exploratory data analysis (EDA)

What are the common observations in the dataset

(Answer in 5 Lines)

* Anomaly detection

What Anomalies can be seen in the dataset

(Answer in 5 Lines)

* Summary statistics

Run code and explain the summary in 5 lines

2. Data Visualization

Run the code to see the below visualizations of the dataset’s Features:

* Histograms
* Scatter plots
* Contour plots
* Matrix plots

Now explain the observations and two pro and two cons of each of the above visualization methods in respect to the given dataset

**Coding Exercise 2: Classification and Evaluation**

**Implement all the below mentioned algorithms on the dataset**

* Nearest-neighbor

NearestNeighborClassifierManual:

Initialize X\_train and y\_train as None

Fit(X\_train, y\_train):

Set X\_train and y\_train to the provided input

Predict(X\_test):

Initialize an empty list for predictions

For each sample x\_test in X\_test:

Calculate the distances between x\_test and all samples in X\_train

Find the index of the nearest neighbor in X\_train

Append the corresponding y\_train label to the predictions list

Return the predictions list

* Gaussian Naïve Bayes

GaussianNaiveBayesClassifierManual:

Initialize class\_priors, class\_means, and class\_variances as None

Fit(X\_train, y\_train):

Initialize dictionaries for class\_priors, class\_means, and class\_variances

For each class c in unique classes of y\_train:

Calculate the class prior probability

Calculate the mean and variance of each feature for the class

Predict(X\_test):

Initialize an empty list for predictions

For each sample x\_test in X\_test:

Initialize an empty list for posteriors

For each class c:

Calculate the likelihood of x\_test belonging to class c

Multiply the prior probability by the likelihood to get the posterior probability

Append the posterior probability to the posteriors list

Find the class with the highest posterior probability and append its index to predictions

Return the predictions list

* Support vector machines

SupportVectorMachineClassifierManual:

Initialize learning\_rate, epochs, weights, and bias

Fit(X\_train, y\_train):

Initialize weights and bias as zeros

Repeat for a specified number of epochs:

For each sample x and corresponding label y in X\_train and y\_train:

If the sample is correctly classified:

Update the weights using the learning rate

Else:

Update the weights and bias using the learning rate and the misclassified sample

Predict(X\_test):

Calculate the dot product of X\_test and weights

Subtract the bias from the result

Return the sign of the result as predictions

* Confusion Matrix

ConfusionMatrix:

Initialize y\_true, y\_pred, n\_classes, and matrix

Compute\_confusion\_matrix():

Initialize a matrix of zeros with dimensions n\_classes x n\_classes

For each pair of true and predicted labels:

Increment the corresponding entry in the matrix

Return the computed matrix

Plot():

Plot the confusion matrix using a heatmap

* Evaluation Metrics

EvaluationMetrics:

Initialize y\_true, y\_pred, confusion\_matrix, and metrics

Compute\_metrics():

Compute true positives, false positives, false negatives, and true negatives

Calculate sensitivity, specificity, false positive rate, false negative rate, precision,

recall, and F1 score

Return a dictionary containing all computed metrics

Formulas to Remember:

1. Nearest Neighbor Classifier Manual:

primarily relies on computing distances between points.

2. Gaussian Naive Bayes Classifier Manual:

- Prior Probability: P(C) = Number of samples in class C / Total number of samples

- Likelihood:

P(X|C) = (1 / sqrt(2 \* pi \* var)) \* exp(-0.5 \* ((x - mean) \*\* 2) / var),

where var is the variance and mean is the mean of the feature values for class C.

3. Support Vector Machine Classifier Manual:

- None, as the focus is on updating weights and bias using gradient descent.

4. Confusion Matrix:

- None, as it primarily involves counting the occurrences of true and predicted labels.

5. Evaluation Metrics:

- Sensitivity (True Positive Rate): TP / (TP + FN)

- Specificity (True Negative Rate): TN / (TN + FP)

- False Positive Rate: FP / (FP + TN)

- False Negative Rate: FN / (FN + TP)

- Precision: TP / (TP + FP)

- Recall: TP / (TP + FN)

- F1 Score: 2 \* (Precision \* Recall) / (Precision + Recall)

Instructions:

1. Implement the above-mentioned algorithms
2. Compare their results of the algorithms with each other
3. Write in a few lines on why the results differ from one another