**Class:** DATS 6202– Machine Learning

**Date:** 4/29/2023

**Team 11**: Amjad Altuwayjiri

**Human Activities with Smartphone Sensors Classification Using Different Machine Learning Network Algorithms**

1. **Introduction**

Human activity recognition using smartphones is a growing area of research with many potential applications, including health monitoring, sleep tracking, and fitness tracking (Chawla, Prakash, & Chawla, 2021; Islam et al., 2022).  In this project, we used the UCI  Human Activity Recognition with Smartphones dataset to train and test machine learning algorithms for accurately classifying different human activities based on accelerometer and gyroscope data collected from a Samsung Galaxy S II smartphone.

Various neural network and machine learning algorithms were applied, including MLP, LVQ, and SVM, to classify the activities and tested their performance using different metrics. Novelty detection was conducted too to detect any unusual behaviors in the dataset and identify any subjects with injuries or health problems.

The analysis results indicate that the SVM model achieved the highest accuracy of 100%, while the One-class SVM revealed no anomalies in the data. Thus, we can conclude that human activities can be accurately classified using the current dataset. However, for more comprehensive insights, a more diverse dataset with varying subject demographics would be beneficial.

1. **Personal Portion of Project Contribution**

This section of the project report discusses my personal contributions to the project, which include implementing and evaluating three machine learning models: Multilayer Perceptron (MLP), Learning Vector Quantization (LVQ), and One-class Support Vector Machine (One-class SVM). For each model, I tuned the hyperparameters using various optimization techniques Randomization and Bayesian. I evaluated their performance using accuracy, confusion matrix, precision, recall, F1-score, ROC, and AUC each model with its suitable method. I plotted the learning curves, confusion matrices and ROC|AUC of each model. In addition, I evaluated the data readiness for modeling, such as the number of NA values, feature types, imbalanced data, outlier detection by clustering. To assess data imbalance, I plotted the portions of the targeted column, and I plotted the raw dataset for a general demonstration.

**2.1. Data Preprocessing**

The first step in the process is to preprocess the data. The “sounds.csv” file is loaded into a Pandas DataFrame and the feature types, data imbalance, and number of NA values are checked. The next step is to detect outliers. DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is used to perform clustering on the preprocessed data. The algorithm finds points that are closely packed together and mark them as core samples, and the remaining points as noise or outliers. In the code provided, the DBSCAN algorithm is applied to the preprocessed data, and the number of clusters and outliers are printed. The outliers are then extracted from the data, and a new dataset is created without the outliers for further analysis.

**2.2. MLP**

The hyperparameters of the MLP classifier are tuned using Bayesian optimization, which tries to find the combination of hyperparameters that maximizes the performance of the model on the training set.

The hyperparameters include the number of hidden layers, the activation function, the solver algorithm, the L2 regularization strength, and the initial learning rate. The hyperparameter search space was defined as follows: the number of hidden layers was sampled uniformly from the range of 1 to 200, the activation function was chosen from a categorical distribution of either ReLU or Tanh, the solver algorithm was also chosen from a categorical distribution of either Adam or L-BFGS, the L2 regularization strength was sampled from a log-uniform distribution ranging from 1e-5 to 1e-3, and the initial learning rate was sampled from a log-uniform distribution ranging from 0.0001 to 0.1.

The best model obtained from the search is then evaluated on the test set. We report the accuracy of the model, which is the proportion of correct predictions made by the model. Furthermore, we analyze the performance of the model by computing the confusion matrix, which shows the number of correct and incorrect predictions for each class. Precision, recall, and  F1-score are also calculated to provide further insights into the model's performance. Finally, learning curves are plotted to analyze the model's performance with different training set sizes, and the mean and standard deviation of the training and test scores are calculated.

**2.3. LVQ**

The LVQ model is trained and evaluated using several techniques. First, a set of hyperparameters is defined for the model, and a random search is used to find the best combination of hyperparameters for the model by maximizing the performance on the training set.

The parameter distributions were defined to search over a range of possible values, including the number of prototypes per class, the random state, and the beta value. The prototypes\_per\_class hyperparameter was varied between 1 and 10, while the random\_state was set to range between 0 and 9. The beta hyperparameter was varied over a set of discrete values, including 0, 25, 5, 75, and 1.

The best model is evaluated on the test set to calculate the accuracy, which measures the proportion of correct predictions. Additionally, the confusion matrix is computed to show the number of correct and incorrect predictions for each class. The precision, recall, and F1-score are also calculated to provide further insights into the model's performance. Finally, learning curves are plotted to analyze the model's performance with different training set sizes, and the mean and standard deviation of the training and test scores are calculated.

**2.4. One-class SVM**

The One-class SVM model is trained and evaluated using various techniques. First, the nu hyperparameter is defined, which controls the proportion of data considered anomalous. In this case, nu is set to 0.1, which means that 10% of the data is considered anomalous. The model is trained on the training set and used to predict anomalies on the testing set. The labels for the testing set are converted to binary, with 1 indicating normal data and -1 indicating anomalous data. The confusion matrix is computed to show the number of true and false positives and negatives. The accuracy of the model is calculated as the proportion of correct predictions made by the model. Finally, the decision function output for the test set is obtained, and the ROC curve and ROC area are computed for each class.

1. **Result**
2. **Conclusion**
3. **Calculation**

**Reference**