Digital Communications

Upon completing this section, you will acquire the skills to work with various fundamental machine learning-based classification techniques. You will be proficient in their application to a provided labeled CSI dataset, with the goal of recognizing human activities. Through this process, you will be able to identify the classification method that provides superior accuracy and lower time complexity. Additionally, by examining the confusion matrices, you will gain insights into which activities exhibit the highest levels of accuracy.

I. Human Activity Sensing

A. *Processed Dataset*

In this part, your task involves dealing with a CSI dataset that has been labeled. For this section, you will need to utilize **Python**, as it conveniently offers all the essential packages and libraries required to accomplish the classification tasks at hand. The provided compressed file, **CSI_Dataset** comprises two directories: one designated for inputs (**data**) and the other for the outputs or labels (**label**). Within each folder, you will find two CSV files. In (**data**) you will see “x_train,” which contains the input data for training, and “x_test”, which is input data for testing the classification model. The second folder, **label**, contains two CSV files. “y_train” is the output or label for training a model, and “y_test” is the label for testing a trained classification model.

- “x_train” comprises 3,977 samples, with each sample having 3 antennas, 30 sub-carriers, and 250 packets. It’s important to note that in this dataset, the number of antennas and sub-carriers has been combined, resulting in a total of 90. Consequently, the data is structured in the shape of (3977, 250, 90). On the other hand, “x_test” has 500 samples and has the shape of (500,250,90).
- The output samples align in number with their respective input samples. However, the output data only consists of scalar values for each sample, lacking the second and third dimensions. Consequently, the shape of “y_train” is (3977,), and the shape of “y_test” is (500,).
- Each element in the output variable “y” is an integer ranging from 1 to 7. Each of these integers corresponds to a distinct category representing a type of human activity. Activities are: 1: Lie down - 2: Fall - 3: Walk - 4: Pick up - 5: Run - 6: Sit down - 7: Stand up

B. *K-Nearest Neighbors*

K-Nearest Neighbors (KNN) is a supervised machine learning algorithm used for classification and regression tasks. It operates by classifying or predicting a data point based on the majority class (for classification) or the average of the nearest neighbors’ values (for regression) in the feature space. The number of nearest neighbors to consider, denoted as “k,” is a hyperparameter that you can adjust. KNN is simple to understand but may be sensitive to the choice of “k” and requires feature scaling. For more in-depth information and resources on the topic, you can refer to KNN Classification with scikit-learn and explore the wealth of materials available on the internet. There are numerous tutorials, articles, courses, and documentation that can provide comprehensive insights and guidance on the subject.

C. *Support Vector Machine*

Support Vector Machine (SVM) is a supervised machine learning algorithm used for classification and regression tasks. SVM seeks to find a hyperplane that best separates data points into distinct classes (in the case of classification) or best fits the data (in the case of regression). It works by maximizing the margin between data points and the separating hyperplane while minimizing classification errors. SVM is effective for high-dimensional data, can handle non-linear problems using kernel functions, and is known for its ability to find robust solutions. For more in-depth information and resources on the topic, you can refer to Support Vector Machines with Scikit-learn Tutorial and explore the wealth of materials available on the internet. There are numerous tutorials, articles, courses, and documentation that can provide comprehensive insights and guidance on the subject.

D. *Random Forest*

Random Forest is a powerful machine-learning algorithm used for classification and regression tasks. Here’s a brief explanation of Random Forest classification:

- Ensemble Learning: Random Forest is an ensemble learning method, meaning it combines the predictions of multiple individual decision trees to make more accurate and robust predictions.
- Decision Trees: At the core of a Random Forest are decision trees, which are used to make binary decisions based on input features.
- Bootstrapping: Random Forest uses a technique called bootstrapping to create multiple random subsets of the training data, which are used to train individual decision trees.
- Feature Randomization: For each decision tree, only a random subset of features is considered when making splits. This reduces overfitting and improves generalization.
- Voting: In classification, Random Forest combines the predictions of individual trees by voting. The class that receives the most votes becomes the predicted class.
- Predictive Power: Random Forest is known for its high predictive power, good generalization to unseen data, and resistance to overfitting.
- Feature Importance: It provides a measure of feature importance, which can be helpful in understanding which features have the most impact on the classification.
- Versatility: Random Forest can be used for both classification and regression tasks, making it a versatile choice for various machine-learning problems.

Random Forest is widely used in practice due to its effectiveness, ease of use, and ability to handle large and complex datasets. It is a popular choice in data science and machine learning for tasks like image classification, natural language processing, and many others. For more in-depth information and resources on the topic, you can refer to Random Forest Classification with Scikit-Learn and explore the wealth of materials available on the internet. There are numerous tutorials, articles, courses, and documentation that can provide comprehensive insights and guidance on the subject.

E. *Questions*

1) Apply the KNN classification method for various values of K, including 15, 10, 5, 2, and 1. In each case, compute the classification accuracy. You can refer to the provided resources to learn how to calculate accuracy. Compare the results in terms of time complexity and accuracy.
2) Train an SVM on the provided dataset without specifying any particular parameters, and subsequently, calculate and report the classification accuracy.

3) Employ the Random Forest classification method with different numbers of estimators. Set the number of estimators to 5, 10, and 20, and determine the classification accuracy for each case. Observe and compare the differences in the results between these scenarios to analyze how the number of estimators affects the classification accuracy.
4) Plot the normalized confusion matrix and analyze your findings for the top-performing scenario identified in the top-performing case in each of the previous questions. Therefore, you must plot 3 different normalized confusion matrices and explain your insights. You may refer to an external resource, such as Confusion matrix or any other available materials on the web, to understand the concept of creating and interpreting a confusion matrix.

F. Tips and Hints

- It's important to handle the data loading process from the CSV files with care. Keep in mind that each sample has 2 dimensions, whereas classification methods typically accept 1-dimensional inputs. Your responsibility is to ensure that the input data is appropriately transformed into 1 dimension to align with the requirements of the classification methods.
- You can implement your solutions by following the guidelines and recommendations provided in the available resources. It's important to use Numpy and Sklearn effectively for your implementation.