School of Science, Computing and Engineering Technologies

COS40007

Artificial Intelligence for Engineering

Portfolio Assessment-2: "Systematic approach to develop ML model"

Studio 3 – Portfolio 3

Students' Name: Duc Thinh Pham - 104169675

Lecturer: Dr. Trung Luu

Date: September 27th, 2024

CONTENTS

1 Introduction 3

2 Studio Work 4

2.1 Summary Table of Studio 3: Activity 6 4

2.2 Summary Table of Studio 3: Activity 7 4

3 Portfolio Work 6

3.1 Data Collection 6

3.1.1 Overview 6

3.1.2 Load & Extraction 6

3.2 Create Composite Columns 7

3.2.1 Composite Features 7

3.2.2 Data Structure after combination 8

3.3 Data Preprocessing and Feature Computation 8

3.3.1 Statistical Features to be Computed: 8

3.4 Training 8

3.4.1 Modeling Steps: 9

3.4.2 Model Evaluation Metrics 9

3.5 Model Selection 9

3.5.1 Summary Table of Portfolio: Activity 5.1 9

3.5.2 Summary Table of Portfolio: Activity 5.2 10

4 Appendix 11

1 INTRODUCTION

In this report, I will present the development process of a machine learning classification model designed to distinguish between two types of meat processing activities: boning and slicing. The dataset used contains acceleration data collected from 17 body-worn sensors, capturing 66 different values across x, y, and z axes. The report outlines a systematic approach, including data pre-processing, feature computation, model training, and evaluation, following the guidelines from previous studio activities. This work aims to apply concepts learned in Week 2 to generate meaningful features and accurately classify activities using various machine learning models such as SVM, SGD, Random Forest, and MLP classifiers.

2 STUDIO WORK

2.1 SUMMARY TABLE OF STUDIO 3: ACTIVITY 6

Model	Train-test	Cross-validation
Original Features	0.888793	0.892164
Hyperparameter Tuning	0.834909	0.8427
Feature Selection + Hyperparameter Tuning	0.8561	0.8560
PCA + Hyperparameter Tuning	0.8469	0.8434

Table 1: SVM Models Accuracy Comparisons with ampc dataset

The evaluation of model performance with the ampc dataset reveals that the original feature set achieves the highest accuracy, with a train-test accuracy of **0.888793** and a cross-validation accuracy of **0.892164**. Hyperparameter tuning results in a slight drop in both train-test (**0.834909**) and cross-validation (**0.8427**) scores, indicating that while it refines performance, it may introduce instability. Feature selection combined with hyperparameter tuning does not surpass the original model's performance, achieving similar scores (**0.8561** for train-test and **0.8560** for cross-validation) and slightly increasing variance. Furthermore, using PCA with hyperparameter tuning drastically reduces accuracy to **0.8469** (train-test) and **0.8434** (cross-validation), highlighting a significant loss of critical information. Overall, the findings suggest that the original feature set provides the best balance of accuracy and stability, with dimensionality reduction techniques like PCA detrimental to model performance.

2.2 SUMMARY TABLE OF STUDIO 3: ACTIVITY 7

Classifier	Train-test	Cross-validation
Best SVM	0.888793	0.892164
Hyperparameter Tuning	0.8839	0.8659
Random Forest Classifier	0.92	0.9265
MLP Classifier	0.8684	0.8474

Table 2: Classifier Accuracy Comparisons with ampc dataset

The Random Forest classifier demonstrates the highest performance among the evaluated models, achieving a train-test accuracy of **0.92** and a cross-validation accuracy of **0.9265**, indicating its strong generalization capability. The Best SVM classifier also performs well, with train-test and cross-validation accuracies of **0.888793** and **0.892164**, respectively. In contrast, hyperparameter tuning of the SVM yields slightly lower performance, while the MLP classifier exhibits the weakest results, with accuracies of

0.8684 for train-test and **0.8474** for cross-validation. Overall, the findings suggest that the Random Forest model is the most effective choice for this dataset, followed closely by the Best SVM.

3 PORTFOLIO WORK

3.1 DATA COLLECTION

3.1.1 Overview

The dataset contains acceleration data collected using 17 body-worn sensors positioned at various locations on the body, resulting in 22 positional values along the x, y, and z axes (66 columns in total). Each row in the dataset represents 1 second of recorded data, with 60 rows equating to 1 minute of data. The dataset captures two different meat processing activities: boning and slicing, labeled as 0 and 1 respectively.

For this assessment, based on my student ID (104169675), only the data corresponding to the *Right Hand (x, y, z)* and *Left Hand (x, y, z)* acceleration data will be utilized. This selection is determined by the last digit of your student ID (5), which designates the use of hand-related sensor data.

Key points:

- Number of Sensors: 17
- Used Data: Right Hand (x, y, z) and Left Hand (x, y, z) columns
- Frame Rate: 1 second per frame, 60 frames per minute
- Activities: Boning (0) and slicing (1)
- Total Columns for Assessment: 8 (Frame, Right Hand (x, y, z), Left Hand (x, y, z), Activity Label)

3.1.2 Load & Extraction

1. **Loading the Dataset**:
    - The dataset comprises two CSV files: Boning.csv and Slicing.csv, which contain acceleration data corresponding to two distinct meat processing activities. We will load these files into Pandas DataFrames for further processing.
2. **Standardizing Column Names**:
    - To ensure consistency and ease of access during data manipulation, we will standardize the column names by:
        - Removing any spaces.
        - Replacing dots with underscores.
        - Converting all text to lowercase.
3. **Extracting Relevant Columns**:
    - Based on the last digit of the student ID (104169674), we will focus on the acceleration data from the **Right Hand** and **Left Hand** sensors. Specifically, we will extract the following columns:
        - Frame (time index)

- Right Hand (x, y, z)
- Left Hand (x, y, z)
- Class (0 for boning, 1 for slicing)

4. **Combining Data**:
    - After extracting the relevant columns from both datasets, we will combine them into a single DataFrame that will serve as the basis for feature extraction and model training.

3.2 CREATE COMPOSITE COLUMNS

In this section, I will create composite columns from the existing acceleration data to derive additional features that may enhance the model's predictive capabilities. The composite features will include root mean square (RMS) values for combinations of acceleration axes as well as roll and pitch angles derived from the acceleration data.

3.2.1 Composite Features

1. **RMS Value of x and y**:

$$RMS_{xy} = \sqrt{\frac{1}{2}(x^2 + y^2)}$$

This value captures the combined magnitude of the x and y accelerations.

2. **RMS Value of y and z**:

$$RMS_{xy} = \sqrt{\frac{1}{2}(y^2 + z^2)}$$

This reflects the combined magnitude of the y and z accelerations.

3. **RMS Value of z and x**:

$$RMS_{xy} = \sqrt{\frac{1}{2}(z^2 + x^2)}$$

This captures the combined magnitude of the z and x accelerations.

4. **RMS Value of x, y, and z**: This provides an overall measure of acceleration across all three axes.

5. **Roll**:

$$Roll = \frac{180}{\pi} \times atan2(y, \sqrt{x^2 + z^2}$$

This angle is calculated to determine the rotation about the x-axis.

6. **Pitch**:

$$Pitch = \frac{180}{\pi} \times atan2(x, \sqrt{y^2 + z^2}$$

This angle is calculated to determine the rotation about the y-axis.

3.2.2 Data Structure after combination

The final dataset will have the following column structure:

- **Column 1**: Frame (time index)
- **Columns 2-4**: Acceleration values for the Right Foot (x, y, z)
- **Columns 5-7**: Acceleration values for the Left Foot (x, y, z)
- **Columns 8-13**: Six computed features (RMS, roll, and pitch) for the Right Foot
- **Columns 14-19**: Six computed features (RMS, roll, and pitch) for the Left Foot
- **Column 20**: Class label (0 for boning, 1 for slicing)

3.3 Data Preprocessing and Feature Computation

In this step, I will compute statistical features for each relevant column in the dataset, specifically for columns 2 to 19, which include both original acceleration values and the computed features for the Right and Left Feet. Each feature will be calculated over a period of one minute, defined as 60 consecutive frames.

3.3.1 Statistical Features to be Computed:

For each of the 18 columns, the following features will be computed:

1. **Mean**: The average value over one minute.
2. **Standard Deviation**: A measure of the variation or dispersion of the values.
3. **Minimum**: The smallest value recorded in the minute.
4. **Maximum**: The largest value recorded in the minute.
5. **Area Under the Curve (AUC)**: This represents the integral of the values over the time period, providing insight into the overall activity level.
6. **Peaks**: The number of peaks in the acceleration data, indicating significant changes in the motion.

After computing these features for each column, we will have an extensive set of features ($18 \times 6 = 108$ in total) derived from the original dataset. These features will significantly enhance our ability to train machine learning models and improve their predictive accuracy regarding the classification of meat processing activities (boning vs. slicing). The next steps will involve further data analysis and the selection of appropriate modeling techniques.

3.4 Training

In this section, I will develop Support Vector Machine (SVM) classifiers using the dataset containing 108 features derived from the previous steps. The modeling process will involve several configurations to ensure robust evaluation and selection of the best model.

3.4.1 Modeling Steps:

1. **Train-Test Split**:
    - The dataset will be divided into training and testing sets in a 70/30 ratio. This means that 70% of the data will be used for training the model, while 30% will be reserved for testing its performance.
2. **10-Fold Cross Validation**:
    - To ensure that our model generalizes well, we will implement 10-fold cross-validation. This technique involves dividing the training data into 10 subsets, training the model on 9 subsets, and validating it on the remaining subset. This process will be repeated 10 times, with each subset serving as the validation set once.
3. **Hyperparameter Tuning**:
    - We will perform hyperparameter tuning to optimize the performance of the SVM classifiers. This involves searching for the best combination of hyperparameters (e.g., kernel type, regularization parameter) using techniques such as grid search or randomized search.
4. **Feature Selection**:
    - In addition to hyperparameter tuning, we will also explore model performance using the 10 best features identified through feature importance techniques (e.g., recursive feature elimination or feature selection based on model coefficients).
5. **Principal Component Analysis (PCA)**:
    - Lastly, we will apply PCA to reduce the dimensionality of the feature set to the 10 principal components, followed by hyperparameter tuning to assess the model's performance with this reduced feature set.

3.4.2 Model Evaluation Metrics

- **Accuracy Score**:
  This metric measures the proportion of correctly classified instances in the test set. The accuracy score provides a straightforward interpretation of the model's performance, indicating how well the model can classify the activities.
- **Cross-Validation**:
  Cross-validation is a technique used to assess how the results of a statistical analysis will generalize to an independent dataset. It involves dividing the training data into multiple subsets (folds), training the model on a portion of the data, and validating it on the remaining portion. The cross-validation score is calculated by averaging the performance metrics (such as accuracy) across all folds, which helps ensure that the model is robust and not overfitting to a specific subset of the data.

3.5 MODEL SELECTION

3.5.1 Summary Table of Portfolio: Activity 5.1

Model	Train-test	Cross-validation
Original Features	0.842105	0.7569
Hyperparameter Tuning	0.7701	0.7569
Feature Selection + Hyperparameter Tuning	0.8116	0.7893
PCA + Hyperparameter Tuning	0.7729	0.7519

Table 3: SVM Models Accuracy Comparisons with ampc2 dataset

The evaluation of classifier performance reveals that the model using original features achieves a train-test accuracy of **0.842105** and a cross-validation accuracy of **0.7569**, serving as a strong baseline. Hyperparameter tuning alone results in a lower train-test accuracy of **0.7701** while maintaining the same cross-validation score, indicating a lack of improvement and potential overfitting. Conversely, the combination of feature selection and hyperparameter tuning enhances accuracy, achieving **0.8116** for train-test and **0.7893** for cross-validation. However, applying PCA with hyperparameter tuning decreases accuracy to **0.7729** for train-test and **0.7519** for cross-validation, suggesting that dimensionality reduction may compromise the model's performance. Overall, these findings underscore the value of feature selection in improving model effectiveness while cautioning against the indiscriminate use of PCA.

3.5.2 Summary Table of Portfolio: Activity 5.2

Classifier	Train-test	Cross-validation
Best SVM	0.888793	0.892164
Hyperparameter Tuning	0.8839	0.8659
Random Forest Classifier	0.92	0.9265
MLP Classifier	0.8684	0.8474

Table 4: Classifier Accuracy Comparisons with ampc2 dataset

The evaluation of classifier performance shows that the **Random Forest Classifier** achieves the highest accuracy, with a train-test score of **0.92** and a cross-validation score of **0.9265**, indicating strong generalization capabilities. The **Best SVM** follows closely, with a train-test accuracy of **0.888793** and cross-validation accuracy of **0.892164**, demonstrating robust performance as well. The **Hyperparameter Tuning** yields slightly lower accuracies at **0.8839** for train-test and **0.8659** for cross-validation, suggesting minimal improvement over the baseline model. Lastly, the **MLP Classifier** presents the weakest results, achieving a train-test accuracy of **0.8684** and a cross-validation accuracy of **0.8474**, indicating that it may not be as effective for this dataset. Overall, the Random Forest Classifier stands out as the most reliable model, while SVM also demonstrates competitive performance.

4 APPENDIX

For submission, I will create a shared folder that includes all data files (including intermediate files) and the associated code, as well as a link to my repository:

My GitHub repository: thinhpham1807/COS40007---Artificial-Intelligence-for-Engineering (github.com)

Data files:
https://drive.google.com/drive/folders/1YKb1ne0l4HChTw7oVUK59oTyg78tcgmy?usp=sharing