Swinburne University of Technology

Faculty of Computer Science, Arificial Intelligence

Portfolio Assessment 2: "Systematic approach to develop Machine Learning model"

Author - ID:
Trung-Hieu Nguyen
103488337

Lecturer:
Dr. Trung Luu

Studio 3 and Portfolio Week 3

Ho Chi Minh, Vietnam

September 27, 2024

Contents

Chapter 1

Introduction

1.1 Motivation

In week 3, we investigated the use of several machine learning models—including Support Vector Machines (SVM), Random Forest (RF), Stochastic Gradient Descent (SGD), and Multilayer Perceptron (MLP)—to classify sensor data from meat processing activities. The data underwent preprocessing and feature engineering to improve model performance. Various models were trained and assessed, with the SVM model being evaluated using different techniques such as hyperparameter tuning, feature selection, principal component analysis (PCA), and composite feature computation. Among these, the SVM model with hyperparameter tuning was often preferred for its simplicity and computational efficiency. RF and MLP were chosen to be the best classifier in different dataset. Despite achieving high accuracy, the potential for overfitting was noted due to outliers, indicating a need for further validation. This problem were identified using cross-validation technique.

Chapter 2

Studio Work

2.1 Summary Table of Studio 3: Activity 6

Model	Train-Test	Cross-Validation
Original Features	0.918601	0.9129 ± 0.0072
With Hyperparameter Tuning	0.921181	0.9111 ± 0.0100
Feature Selection + Hyperparameter Tuning	0.918028	0.9106 ± 0.0124
PCA + Hyperparameter Tuning	0.892519	0.8902 ± 0.0094

Table 2.1: Model Accuracy Comparisons with Dataset 1

Hyperparameter tuning slightly improves train-test accuracy (from 0.9186 to 0.9212), but does not significantly enhance cross-validation performance (drop to 0.9111), and might make the model less stable (± 0.0100). Feature selection combined with hyperparameter tuning doesn't yield better results than the original model and slightly increases variance. PCA drastically reduces the accuracy (0.8925 for train-test split and 0.8902 for cross-validation), indicating that dimensionality reduction via PCA does not improve this model's performance and may lead to loss of important information. In conclusion, the original feature set without any dimensionality reduction seems to yield the best balance between accuracy and stability, with hyperparameter tuning offering only marginal improvements.

2.2 Summary Table of Studio 3: Activity 7

The Random Forest classifier performs the best overall, with the highest accuracy on both the train-test (0.9286) and cross-validation (0.9184) metrics. Both Best SVM and MLP demonstrate good generalization with minimal differences between train-test

Classifier	**Train-Test**	**Cross-Validation**
Best SVM	0.918601	0.9129
Stochastic Gradient Descent	0.882201	0.8850
Random Forest	0.928632	0.9184
Multilayer-Perceptron	0.893665	0.89496

Table 2.2: Classifier Accuracy Comparisons with Dataset 1

and cross-validation scores. MLP shows the least overfitting, as its cross-validation accuracy slightly exceeds its train-test score. SGD has the lowest performance across both metrics, although its close cross-validation score shows that it generalizes reasonably well despite being less accurate.

Chapter 3

Portfolio Work

3.1 Data Collection

3.1.1 Overview

This study uses a dataset containing acceleration data collected from 17 body-worn sensors placed at various positions on the body. These sensors capture acceleration values along the x, y, and z axes, resulting in a total of 66 columns of data (22 body positions × 3 axes) in each file. The dataset includes two separate files, each corresponding to a different meat processing activity: boning and slicing. Each row in these files represents data from a 1-second frame, meaning 60 frames provide data for one minute.

For this assessment, not all sensor data will be used. Based on my student ID (103488337), the last digit (7) designates the columns corresponding to the Right Foot (x, y, z) and Left Foot (x, y, z) acceleration data for analysis. Therefore, the relevant data for this task will consist of:

- The *frame* column, which denotes the time index.
- Six columns representing the acceleration values for the right and left foot along the x, y, and z axes.
- A *class* column indicating the activity type: 0 for boning and 1 for slicing.

3.1.2 Load & Extraction

The dataset was processed using the following steps:

1. **Loading Data:** The two CSV files (*Boning.csv* and *Slicing.csv*) were loaded into dataframes. The data was cleaned by standardizing column names (removing spaces, replacing dots with underscores, and converting all text to lowercase) to ensure consistency and ease of access during extraction.

2. **Combining Data:** The two datasets were concatenated into a single dataframe for uniformity, with the activity type labeled as either boning (0) or slicing (1) in a new class column. This allows both activities to be analyzed within the same dataframe.

3. **Handling Missing Values:** Missing data in the acceleration columns were detected and imputed using the column mean, ensuring that any missing or incomplete sensor readings did not negatively impact model performance. However, there is no missing value within the original dataset.

4. **Column Selection:** Only the relevant columns:

   - the *frame* column (time index).
   - the *right foot (x, y, z)*, *left foot (x, y, z)* column, which provides coordinates of sensors at sample's feet.
   - the *class* column, which indicates the meat processing activity.

   were extracted for further analysis. The code dynamically identifies and selects the appropriate columns based on the sensor position assigned to the student ID.

3.2 Create Composite Columns

In this section, I generate several composite data points from the acceleration values in the dataset. These composite features offer additional insights by combining the x, y, and z acceleration values into more complex metrics. The following composite columns were created for both the **Right Foot (x, y, z)** and **Left Foot (x, y, z)** data, resulting in 12 new columns (6 for each foot) being added to the dataset.

3.2.1 Composite Features

The composite features created are as follows:

1. **Root Mean Square (RMS) Value of x and y**:

$$\text{RMS}_{xy} = \sqrt{\frac{x^2 + y^2}{2}}$$

This feature provides an aggregate measure of acceleration between the x and y axes.

2. **Root Mean Square (RMS) Value of y and z**:

$$\text{RMS}_{yz} = \sqrt{\frac{y^2 + z^2}{2}}$$

Represents a composite acceleration measure across the y and z axes.

3. **Root Mean Square (RMS) Value of z and x**:

$$\text{RMS}_{zx} = \sqrt{\frac{z^2 + x^2}{2}}$$

Combines the acceleration values between the z and x axes into a single value.

4. **Root Mean Square (RMS) Value of x, y, and z**:

$$\text{RMS}_{xyz} = \sqrt{\frac{x^2 + y^2 + z^2}{3}}$$

A composite value representing the overall acceleration magnitude across all three axes.

5. **Roll**:

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

The roll angle represents the rotation about the front-to-back axis of the body.

6. **Pitch**:

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

The pitch angle describes the rotation about the side-to-side axis of the body.

3.2.2 Data Structure

After adding these new composite features, the dataset expanded to a total of 20 columns:

- **Column 1**: `frame` (time index)
- **Columns 2-4**: Original acceleration values for **Right Foot (x, y, z)**
- **Columns 5-7**: Original acceleration values for **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` (0 for boning, 1 for slicing)

As depicted in Figure 3.1, the addition of composite columns has generated new features that offer deeper insights into the data, which could enhance the performance of the classification model. The visualizations suggest that these features display different degrees of class separability, helping to inform the feature selection process in the following stages.

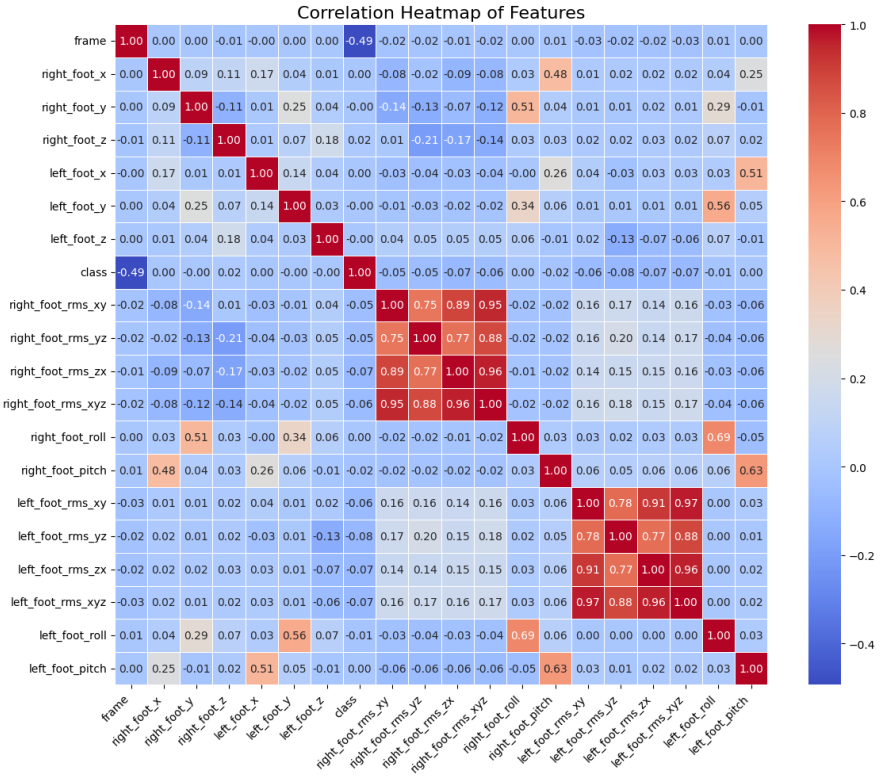

Figure 3.1: Correlation heatmap after the addition of composite features

3.3 Data Pre-processing and Feature Computation

In this step, statistical features are computed for each minute of data. Since each minute corresponds to 60 consecutive frames, features are extracted over a window of 60 frames for each of the 18 relevant columns (columns 2-19 of the dataset, including both the original and composite features). These features are calculated to summarize the behavior of the data in each window, providing a richer feature set for analysis.

3.3.1 Computed Features

For each of the 18 columns, the following six statistical features are calculated:

1. **Mean**: The average value of the data within each window.
2. **Standard Deviation**: A measure of how spread out the values are within the window.
3. **Minimum**: The smallest value observed within the window.
4. **Maximum**: The largest value observed within the window.
5. **Area Under the Curve (AUC)**: The integral of the signal over time, computed using the trapezoidal rule.
6. **Number of Peaks**: The count of local maxima (peaks) within the window, indicating significant oscillations or changes in the signal.

By calculating these six features for each of the 18 columns, a total of $18 \times 6 = 108$ features are generated for each minute of data. These features capture key characteristics of the acceleration data over time, which are essential for improving classification performance. The computation phase effectively converted the raw data into a well-organized format with aggregated features that encapsulate the key attributes of the sensor data. These features will be crucial for the next phase, where we intend to develop a robust classifier to differentiate between boning and slicing activities.

3.4 Training and Evaluation

In this section, we outline the process for training Support Vector Machine (SVM) classifiers and evaluating their performance. The dataset now contains 108 features, which will be used to train and assess models through various methodologies.

3.4.1 Model Evaluation Metrics

To evaluate the performance of the models, we use the following metrics:

- **Accuracy Score**: Measures the proportion of correctly classified instances.
- **Cross-Validation**: Provides an estimate of the model's performance on unseen data by partitioning the dataset into multiple folds.

3.4.2 Experiment Settings

Setting 1: Train-Test Split (70/30)

We first split the dataset into training and test sets with a 70/30 ratio. This allows the model to be trained on a substantial portion of the data and evaluated on a separate, unseen subset. The model's performance is assessed based on accuracy and cross-validation scores.

Setting 2: Hyperparameter Tuning with Train-Test Split

In this setting, we perform hyperparameter tuning to optimize the SVM model's performance. GridSearchCV is used to explore a range of hyperparameters, including the regularization parameter C and the kernel coefficient γ. After tuning, the performance of the best model is evaluated on the test set and compared to the initial model.

Setting 3: Feature Selection with Top 10 Features

Here, we focus on selecting the top 10 most relevant features using statistical methods. This step reduces the feature space to the most impactful attributes, potentially improving model performance by eliminating noise and irrelevant features. The model is retrained and evaluated using these selected features.

Setting 4: Dimensionality Reduction with PCA

Finally, Principal Component Analysis (PCA) is applied to reduce the dimensionality of the feature space to 10 principal components. PCA transforms the feature set into a smaller number of orthogonal components that capture the most variance in the data. The performance of the model with the reduced feature set is then evaluated to determine if dimensionality reduction enhances or maintains model effectiveness.

3.5 Model Selection

3.5.1 Summary Table of Portfolio: Activity 5.1

Classifier	Train-Test	Cross-Validation
Orginal data	0.911357	0.891967
With Hyperparameter Tuning	0.930748	0.911336
Feature Selection + Hyperparameter tuning	0.922438	0.916892
PCA + Hyperparameter tuning	0.897507	0.891967

Table 3.1: Model Accuracy Comparisons using Dataset 2

Hyperparameter Tuning provides the most substantial improvement in both train-

ing and cross-validation accuracy compared to the original model. Feature Selection combined with hyperparameter tuning offers a balance between model complexity and performance, slightly improving cross-validation accuracy. PCA reduces the model's performance, indicating that dimensionality reduction to 10 principal components might not be optimal for this dataset.

3.5.2 Summary Table of Portfolio: Activity 5.2

Classifier	Train-Test	Cross-Validation
Best SVM	0.930748	0.911336
Stochastic Gradient Descent	0.786704	0.869670
Random Forest	0.897507	0.894745
Multilayer-Perceptron	0.936288	0.891892

Table 3.2: Classifier Accuracy Comparisons using Dataset 2

Best SVM and Multilayer-Perceptron offer the highest training accuracies, with the SVM model also demonstrating superior cross-validation performance, making it the most balanced choice for both fitting and generalization. Random Forest provides a consistent performance across both metrics, indicating robustness but slightly lower accuracy compared to the top performers. Stochastic Gradient Descent performs the weakest, particularly in training accuracy, highlighting its limitations in fitting the dataset effectively.

3.5.3 Conclusion

Which SVM model? The SVM with Hyperparameter tuning model is the best SVM model for the problem. The Best SVM model achieves the highest accuracy on the test set and performs well on cross-validation. Its high train-test accuracy indicates effective fitting to the training data, while its strong cross-validation accuracy shows that it generalizes well to unseen data. This balance between high training and validation performance makes it the most suitable SVM model for your problem.

Which Classifier? The Multilayer-Perceptron (MLP) model is the best ML model for the problem based on training performance. The MLP model exhibits the highest training accuracy among the models, demonstrating its ability to fit the training data exceptionally well. Although its cross-validation accuracy is slightly lower compared to the Best SVM, the high training accuracy indicates that the MLP model is effective in learning from the data. If the focus is on achieving the best performance on training data while still maintaining competitive generalization, the MLP model is the best choice.

However, if overall generalization is prioritized, the Best SVM may still be preferred due to its superior cross-validation performance.

Chapter 4

Appendix

For marking and reference purposes, I provide a link to my GitHub repository, which contains the portfolio requirements and my source code, dataset, and results in *.csv* format.

- **GitHub Repository**: link
- **Dataset & Results**: link

Bibliography