BSIT 41043 - Data Mining Data Warehousing

Assignment 02

BSIT 211007 – Sameera Pramodi Jayakody

BCI Campus - Negombo

Abstract

The report evaluates the three classification algorithms performance. Naive Bayes, Neural Network, and Support Vector Machine (SVM) are the three algorithms discuss on a mushroom classification dataset using WEKA. The dataset consists 23 attributes and 8124 instances, was preprocessed by removing irrelevant attributes, handling missing values, and standardizing data. Classification accuracy, parameter effects, and the impact of attribute removal were analyzed. Results indicate varying performance across algorithms, with recommendations for optimization to enhance classification accuracy.

Keywords: Naive Bayes, Neural Network, Support Vector Machine (SVM)

Table of Contents

Table of Diagrams

1. Introduction

In this practical, we will compare the performance of various classification algorithms on the Mushroom Classification Dataset using Weka. The Mushroom Classification Dataset contains descriptions of 23 species of gilled mushrooms from the Agaricus and Lepiota families. Each species classified as either edible, poisonous, or of unknown edibility (combined with poisonous species). The goal is to predict the edibility of mushrooms based on their attributes.

We will compare the following classifiers:

- Naive Bayes: A probabilistic classifier based on Bayes' theorem.
- Neural Network: A multi-layer perceptron (MLP) network for classification.
- SVM (Support Vector Machine): A supervised learning model for classification.

The tasks performed:

- Compare the classification accuracy using Naive Bayes, Neural Network, and Support Vector Machine (SVM). Discuss how altering parameters affects the classification accuracy for each algorithm.
- Identify three attributes that do not contribute to the enhancement of classification accuracy.
- Compare the accuracy of the algorithms after removing the identified attributes and discuss whether the removal of these attributes is justified based on the outcomes.

The objective is to optimize the classification process and determine which algorithm is most effective in predicting mushroom edibility.

2. The Mushroom Classification Dataset

The Mushroom Classification Dataset contains 8,124 instances of 23 mushroom species from the Agaricus and Lepiota families. Each mushroom is described using attributes such as cap shape, color, odor, gill spacing, stalk features, and more. The goal is to classify mushrooms into three categories:

- Edible
- Poisonous
- Unknown (combined with poisonous)

The dataset is used for classification tasks, where the aim is to predict the edibility of a mushroom based on its features.

3. Steps in the Practical

3.1 Loading the Data into Weka.

Open Weka and load the Mushroom Classification Dataset in the Explorer tab in CSV format.

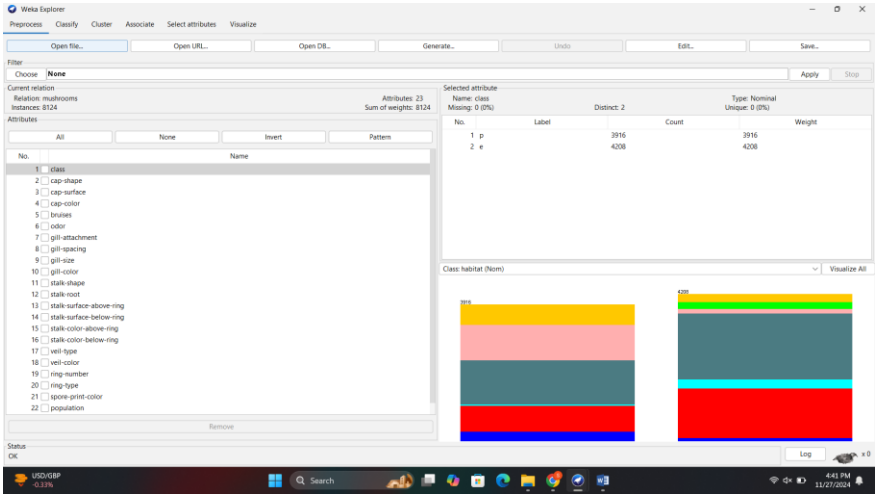

Figure 1: Load the dataset to WEKA

3.2 The Classification algorithms

To compare the classification accuracy of Naive Bayes, Neural Network, and Support Vector Machine (SVM) algorithms on the Mushroom Classification Dataset, first test three algorithms before parameter tuning. Accuracy of each display below.

Naive Bayes: A probabilistic classifier, feature independence, works with for large datasets.

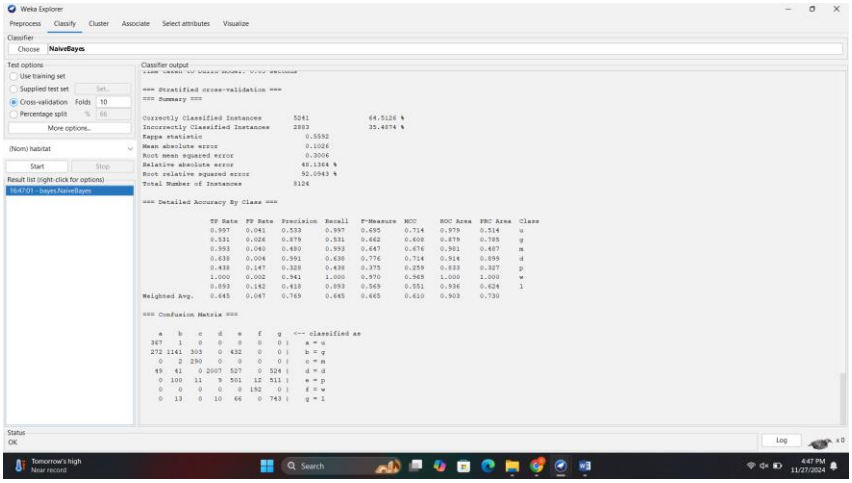

Figure 2: Naive Bayes

Support Vector Machine (SVM): Finds the optimal boundary to separate classes, can handle both linear and nonlinear classification tasks.

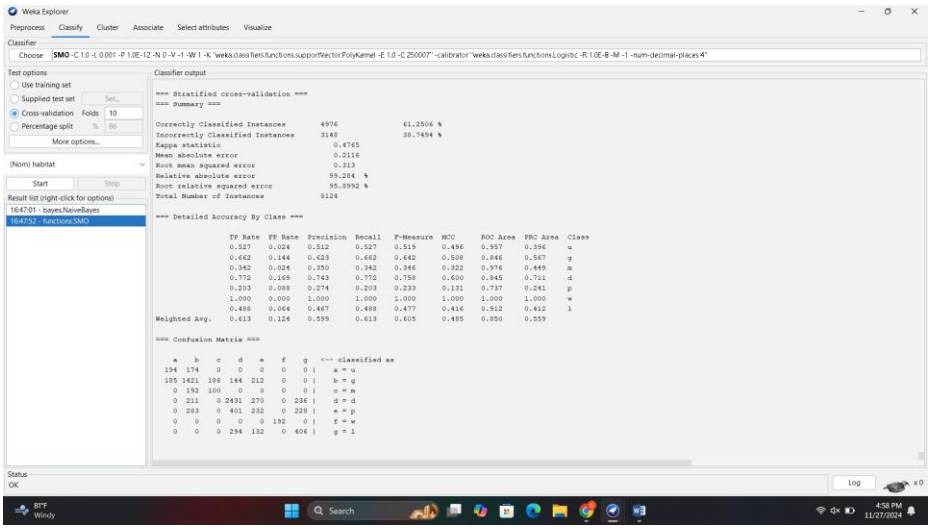

Figure 3: SMO (Sequential Minimal Optimization, WEKA's SVM implementation).

Neural Network: A complex model inspired by the brain and consisting of layers of neurons.

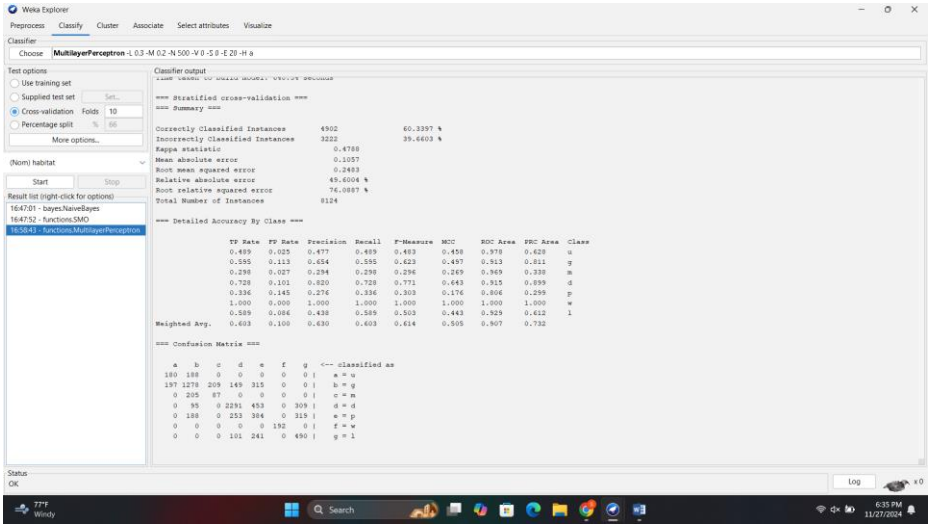

Figure 4: MultilayerPerceptron (WEKA's implementation of a neural network).

Accuracy of classification algorithms. Naïve Bayes has the highest accuracy as 64.52%.

Algorithm	Accuracy
Naive Bayes	64.52%
SMO	61.25%
MultilayerPerceptron	60.33%

Figure 5: Accuracy of algorithms before parameter tuning

3.3 Preprocessing the Data

Before applying the classifiers, processing data is important. Here are the steps followed before evaluate Naive Bayes, Neural Network, and Support Vector Machine (SVM) algorithms.

- Impute missing values using the ReplaceMissingValues filter.

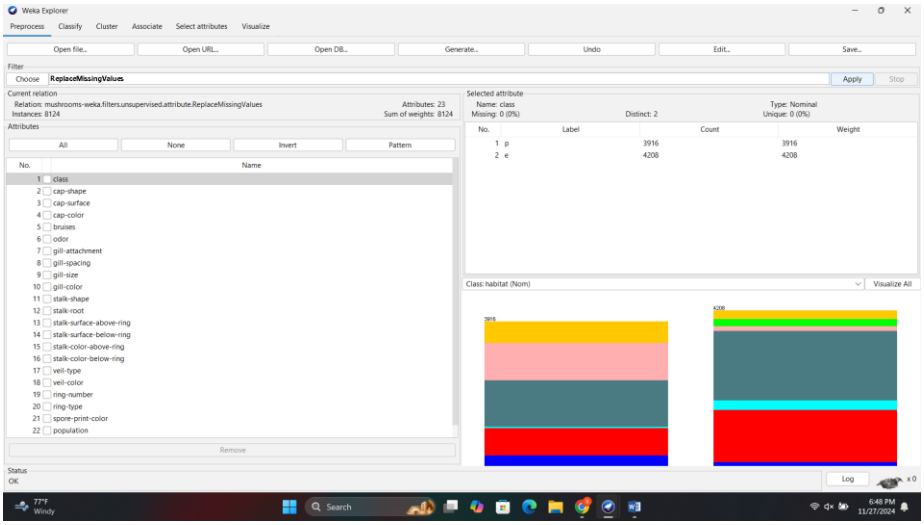

Figure 6: Replace missing values

- Normalization has been applied to the dataset. Applied normalize filter to work with Neural Network algorithm. Normalization helps ensure that all features are on a comparable scale, better performance and leading to faster training,

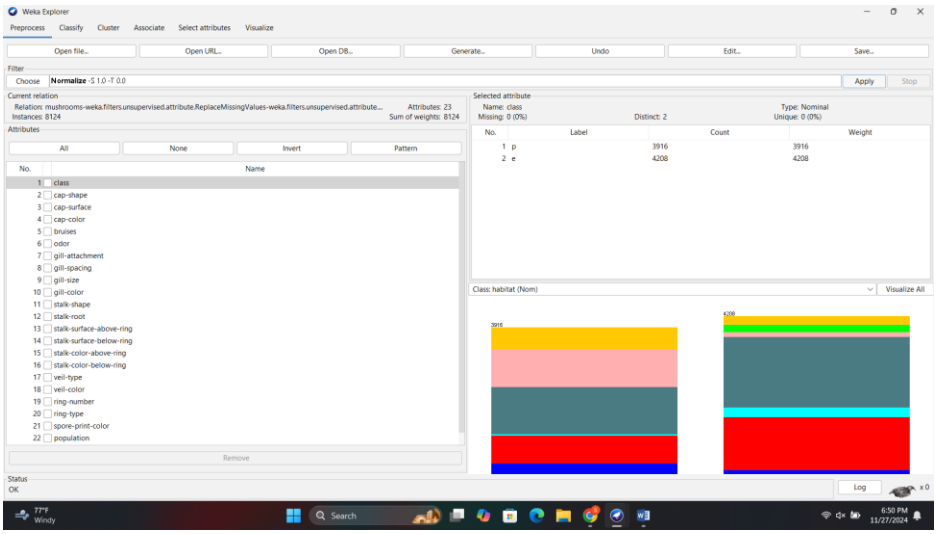

Figure 7: Normalize the dataset

3.4 Setting up Classification Algorithms before Parameter Tuning

- Run Support Vector Machine (SVM) - Accuracy: 61.25%.

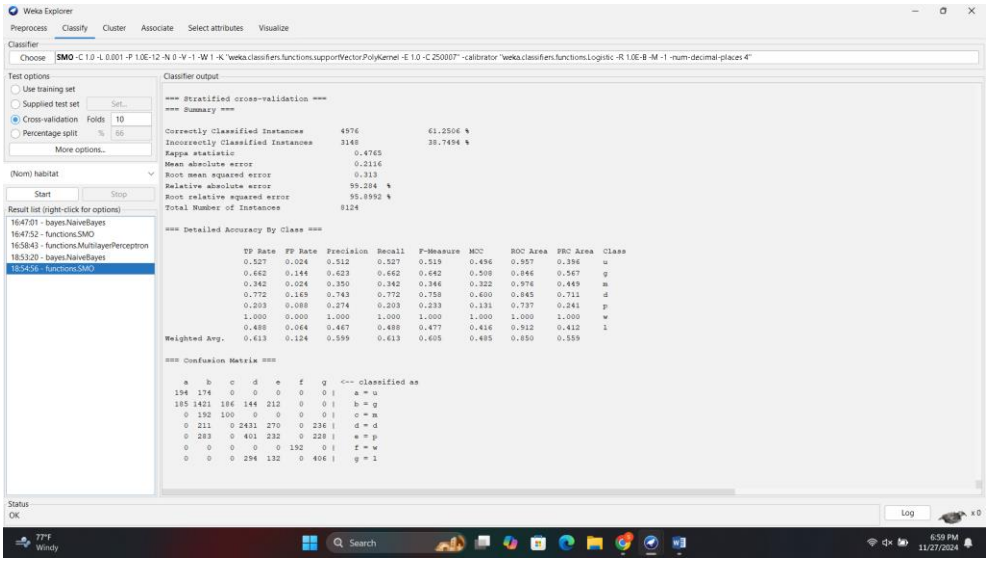

Figure 8: Apply SMO

- Run Neural Network - Accuracy: 60.72%.

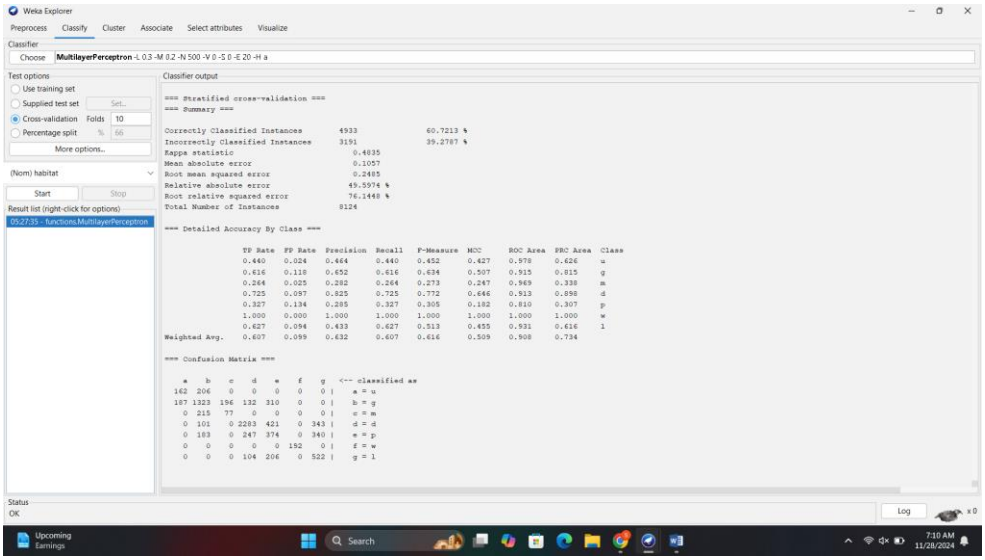

Figure 9: Apply MultilayerPerception

- Run NaiveBayes - Accuracy: 64.43%.

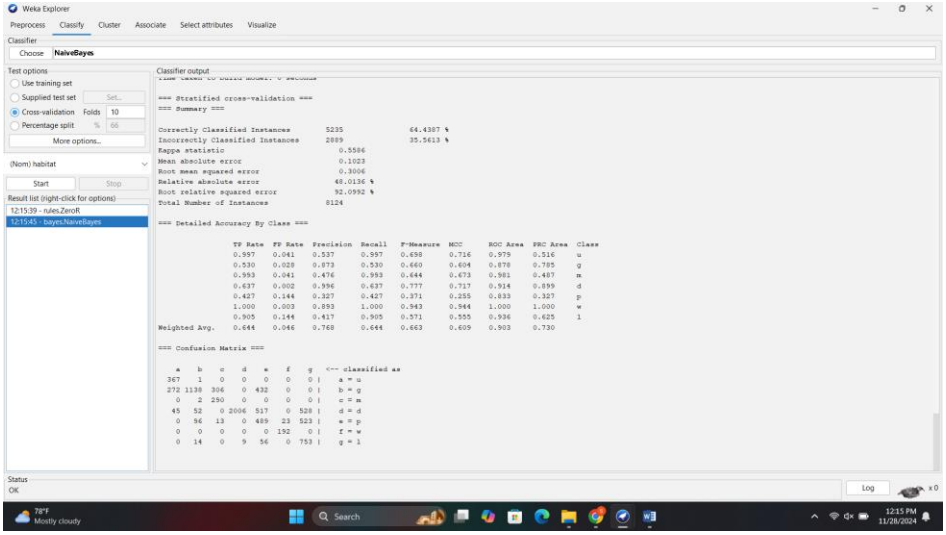

Figure 10: Apply NaiveBayes

3.5 Comparison of Classification Accuracy

Naive Bayes (NB): Accuracy = 64.43%

This model has the highest classification accuracy among the three, it has the best performance overall. It assumes independence between the features, which may have worked well on this data set.

Support Vector Machine (SVM): Accuracy = 61.25%

SVM comes second in terms of accuracy. Though still relatively high, it's a bit lower than Naive Bayes.

Neural Network (NN): Accuracy = 60.72%

Neural Network has the lowest accuracy among the three models. The SVM has the best performance, just beating the neural network by 0.53% differences in accuracy. Therefore, this model is slightly better since it finds a much more general separating boundary between the classes.

3.6 Tuning Parameters and Comparison

Here compare the classification algorithms before parameter tuning and after parameter tuning. Identify how parameters affects to the algorithm's accuracy.

3.6.1 Key Parameters in SMO

- Kernel Type

In the current classification results (before parameter tuning) with the Polynomial Kernel in the SMO (SVM) algorithm have the accuracy: 61.25%. (mention in figure 3 and 5).

By applying the RBF kernel, it achieved higher accuracy 65.47% compared to the Polynomial kernel 61.25%.

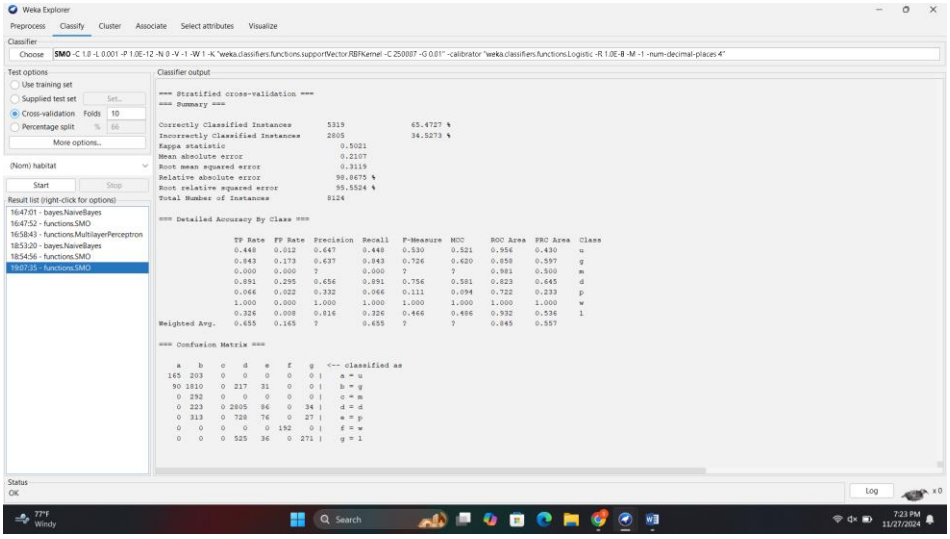

Figure 11: Apply RBF kernel

This suggests that the RBF kernel is more suitable for the dataset. This means SVM with RBF kernel capture more complex relationships in the data.

- C Parameter (Complexity Constant)

When the C value was changed from 1.0 to 0.1 (Lower C value), the correctly classified instances decreased from 65.47% (5319 instances) to 64.34% (5227 instances).

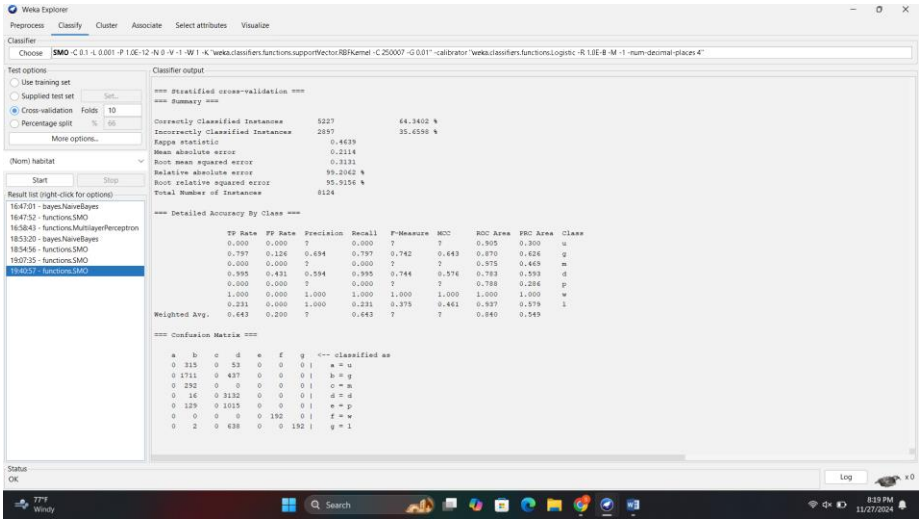

Figure 12: change to Lower C value

When the C value was changed to 10 (Higher C value)

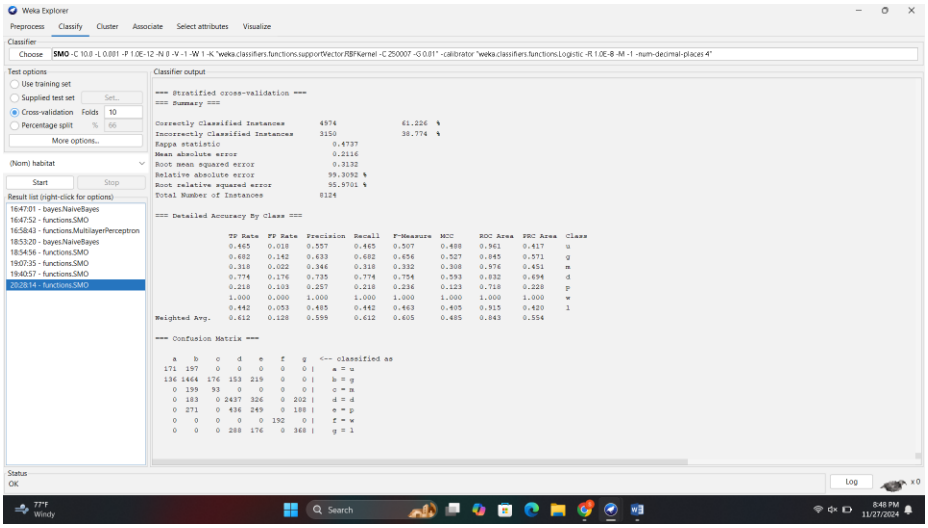

Figure 13: Change to Upper C value

SVM performance with different C values,

C = 1.0 (Default) - Best accuracy at 65.47% and balanced performance.

C = 0.1 (Lower) - Accuracy decreases to 64.34%.

C = 10.0 (Higher) - Accuracy drops further to 61.23% and indicating overfitting, the model focuses too much on the training data.

- Tolerance Parameter (Epsilon)

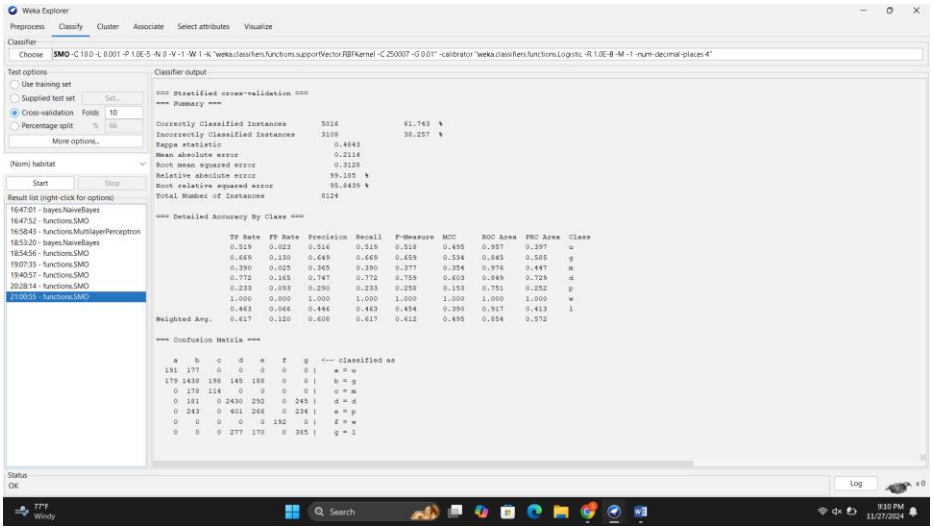

Figure 14: Apply Lower Tolerance with value 1.0E-5

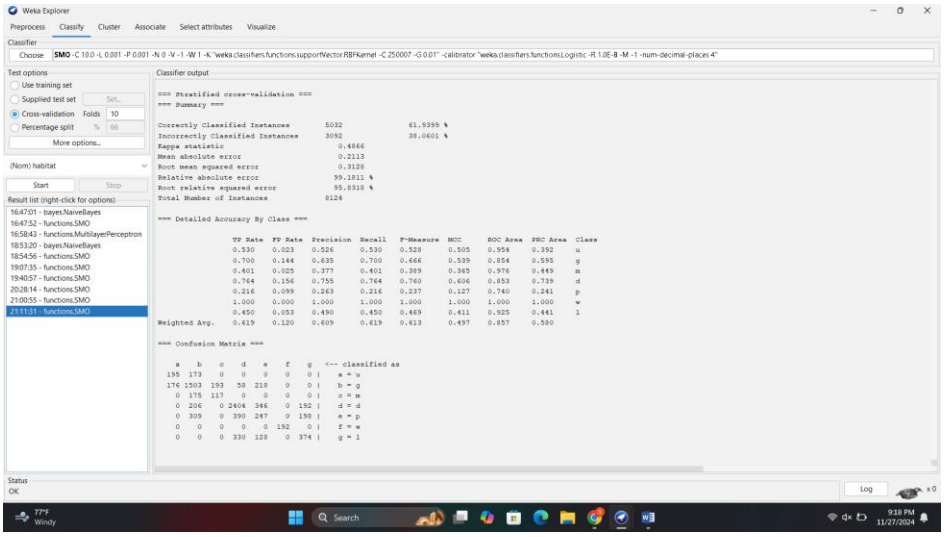

Figure 15: Apply Upper Tolerence with value 1

Comparison of Results for Lower and Upper Epsilon Values,

Lower Epsilon: 61.743% (5016 instances)

Upper Epsilon: 61.9399% (5032 instances)

The upper epsilon value has higher classification accuracy 0.196% and slight improvement in classification accuracy with a higher epsilon.

3.6.2 Key Parameters in MultilayerPerception

- Hidden Layers

By default, Weka uses “a” which is automatically determines the number of neurons per layer. By applying 3 hidden layers, slightly improve model’s accuracy from 61.94% to 64.02%.

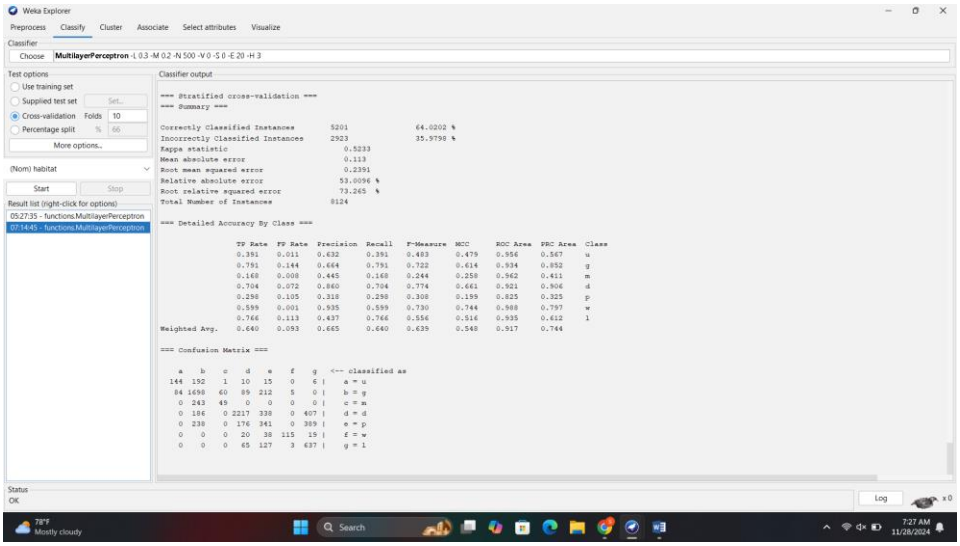

Figure 16: Applying hidden layer as 3

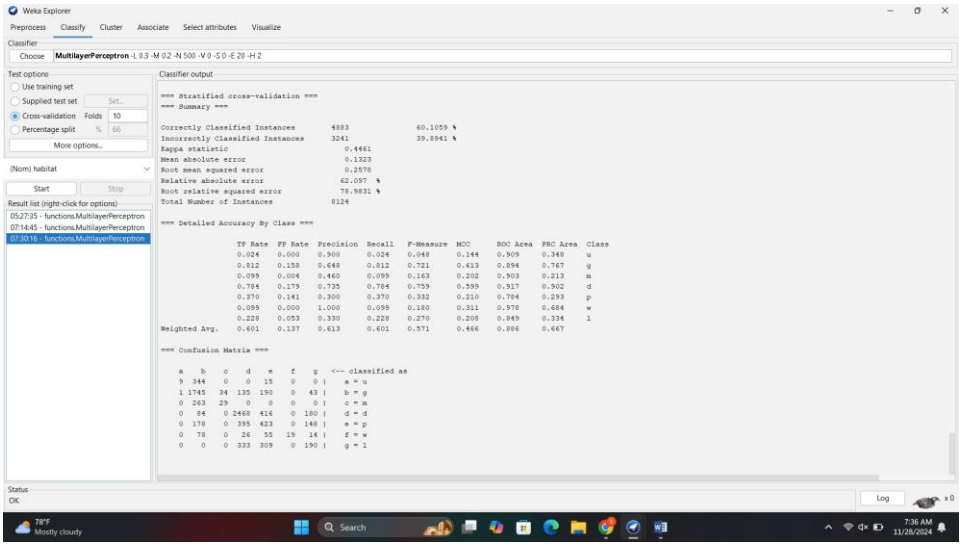

Figure 17: Apply hidden layer as 2

Accuracy: Correctly classified instances dropped to 60.11% when applying 2 hidden layers, which is lower than the previous configuration with 3 hidden layers 64.02%.

- Learning Rate

When comparing the default learning rate 0.3 with the applied learning rate 0.1. It improved overall classification accuracy from 64.02% to 65.74%

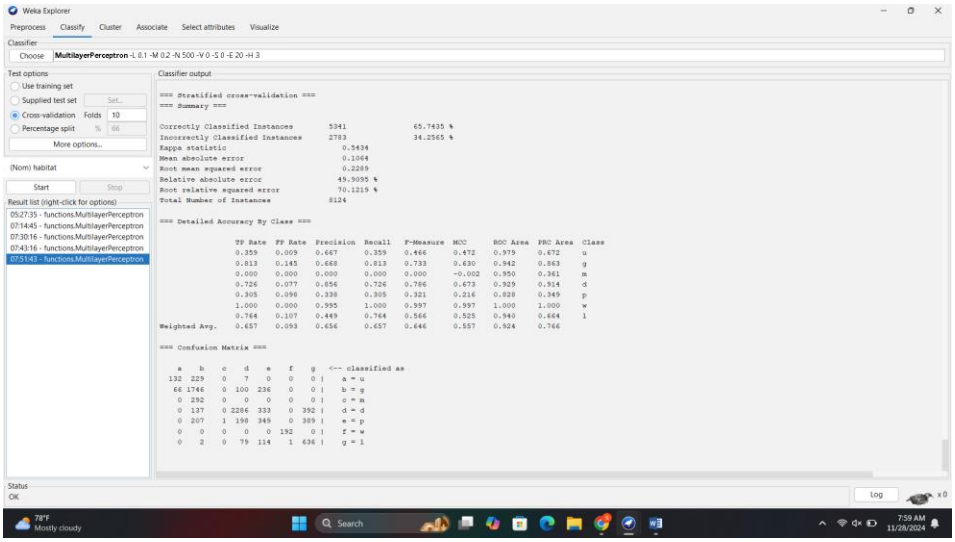

Figure 18: Applying Low learning rate as 0.1

When comparing learning rates (0.3, 0.1, and 0.5); The default learning rate 0.3 achieved accuracy of 64.02% and learning rate 0.1 slightly improved it to 65.74% while learning rate 0.5 maintained a comparable 64.35% accuracy.

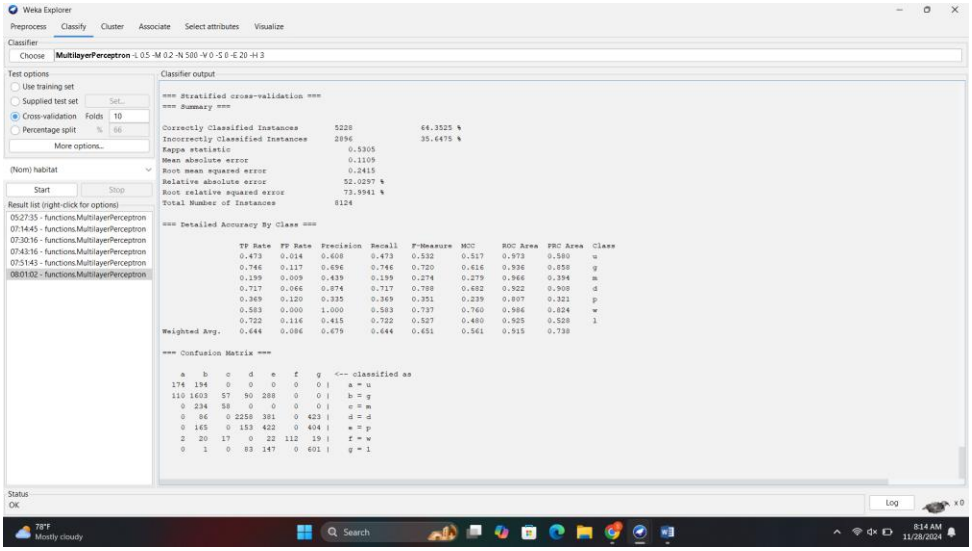

Figure 19: Applying high learning rate as 0.5

- Momentum

The default momentum value is 0.2. When the momentum was increased to 0.9, the accuracy of Neural Network decreased by 4.58% (from 65.74% to 61.16%).

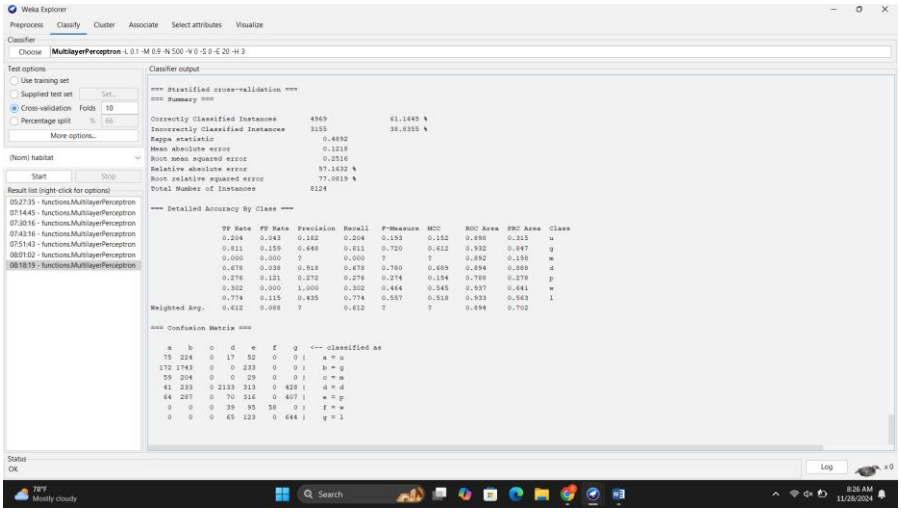

Figure 20: Apply higher momentum value as 0.9

When reducing momentum to 0.1 and accuracy improved to 65.92%.

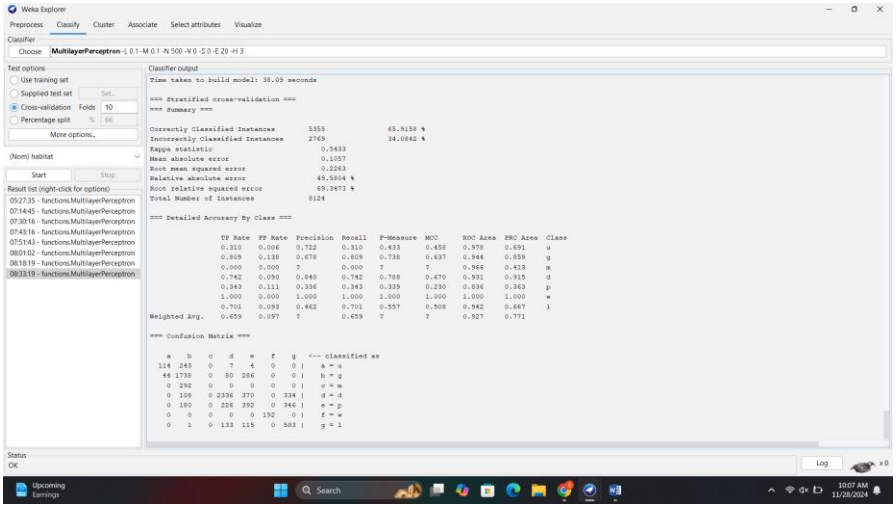

Figure 21: Apply Lower momentum value as 0.1

Here is a detailed accuracy of neural network with momentum parameter.

Momentum value	Accuracy
0.2 (Default)	65.74%
0.9 (Higher)	61.16%
0.1 (Smaller)	65.92%

Figure 22: Accuracies vary in momentum

- Batch Size

The batch size 32 gives higher accuracy 65.92% compared to the default batch size 100 and accuracy 65.74%. Smaller batches allow the model to generalize better on this dataset.

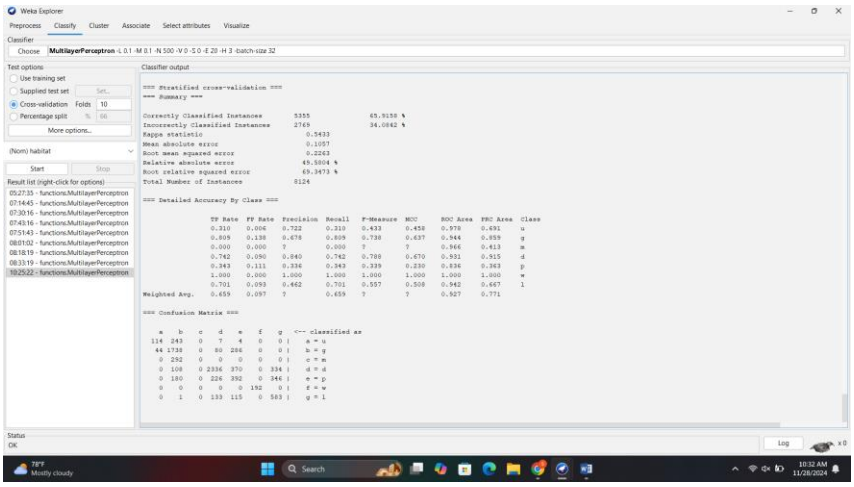

Figure 23: Apply small batch size as 32

3.6.3 Key Parameters in NaiveBayes

- useKernelEstimator

No significant change in accuracy by applying useKernelEstimator

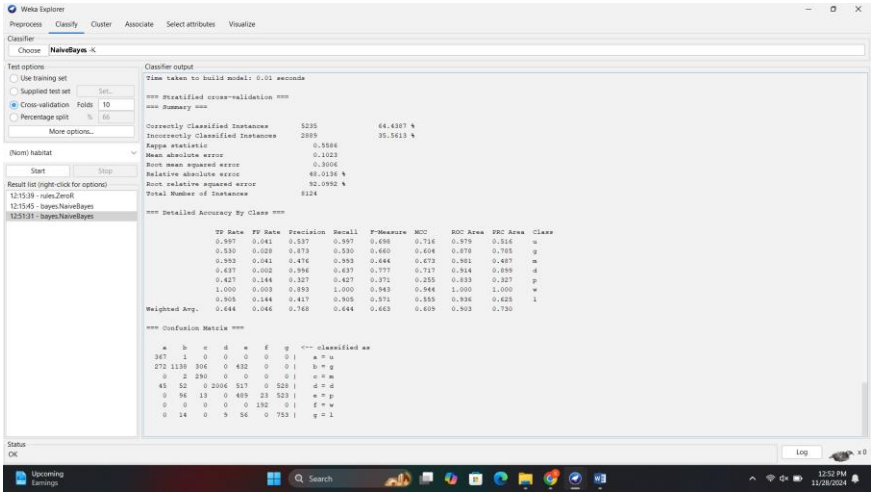

Figure 24: Enable userKernalEstimator as True

- useSupervisedDiscretization

The 64.43% accuracy reflects a no improvement.

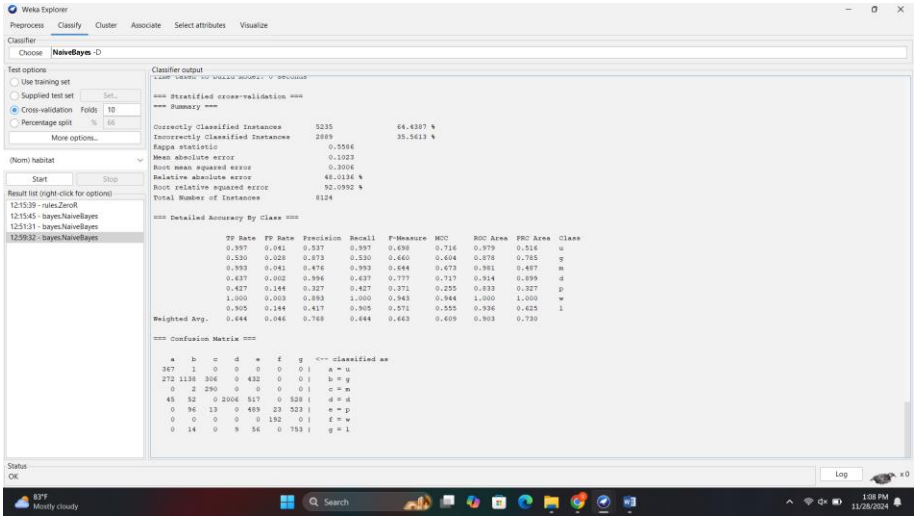

Figure 25: useSupervisedDiscretization as true

- Batchsize

The 64.43% accuracy reflects a moderate improvement.

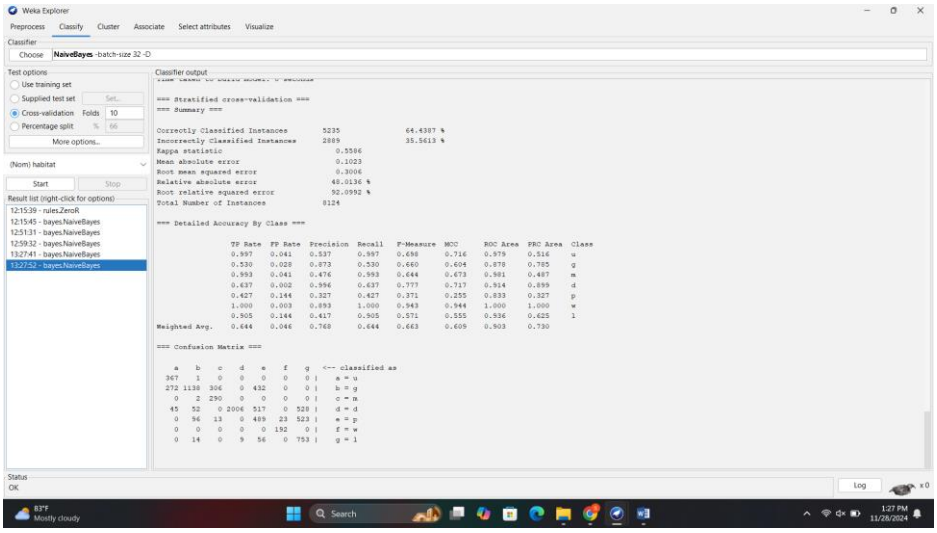

Figure 26: Change the Batchsize as 32

3.7 Identification of three attributes that do not contribute to the classification accuracy enhancement.

From the ranked attributes based on the InformationGainRanking Filter, can identify three attributes that do not contribute to classification accuracy.

- Veil-type (Ranked last, Information Gain: 0.0000)
- Gill-attachment (Information Gain: 0.0736)
- Veil-color (Information Gain: 0.0852)

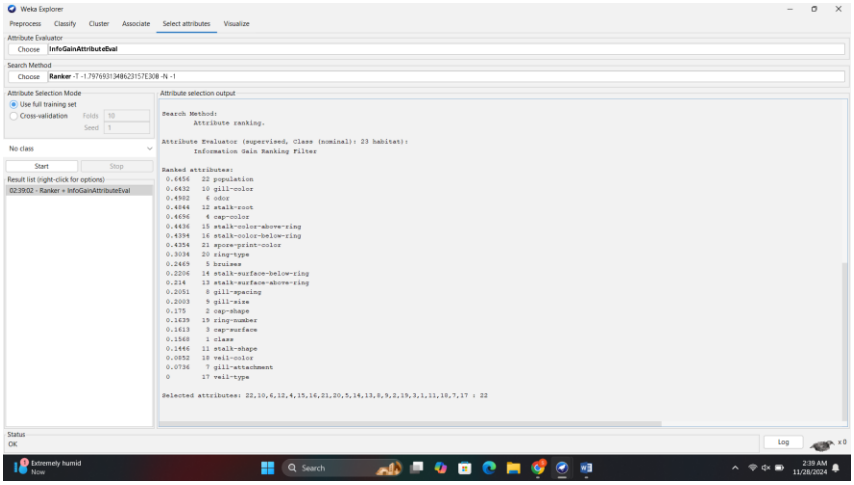

Figure 27: Apply Information Gain Ranking Filter

The three attributes have the lowest scores, can remove from the dataset.

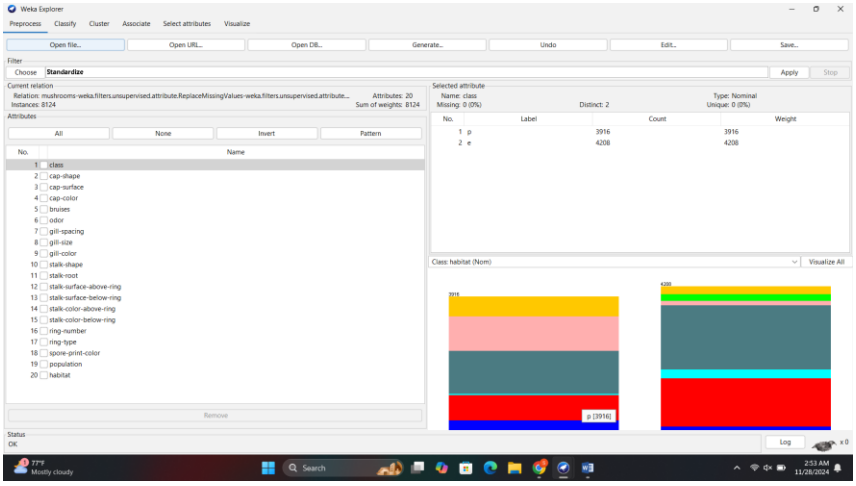

Figure 28: Remove the attributes

3.8 Compare the accuracy of the algorithms after removing the identified attributes

3.8.1 Re-run Classification Algorithms

Apply SMO with the updated dataset

Here's a comparison between the apply SMO with before removing attributes in the dataset and after removing the identified attributes from the dataset (veil-type, gill-attachment, and veil-color).

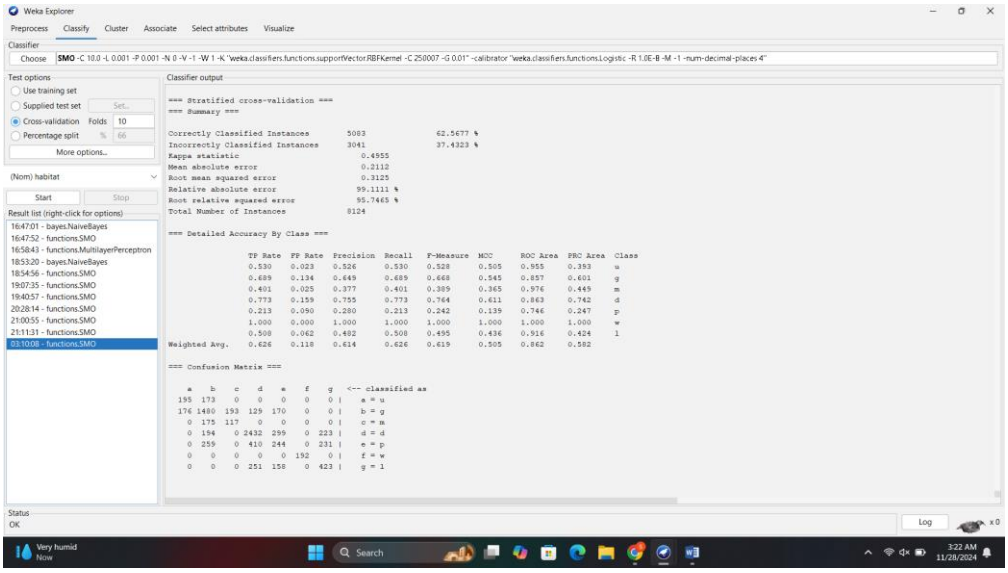

Figure 29: After removing attributes and apply SVM

Identified key observation in Accuracy improvement is,

Correctly classified instances increased by 0.63% (from 61.94% to 62.57%) and Incorrectly classified instances reduced by 0.63%.

Apply MultilayerPerception with the updated dataset

Here's a comparison between the apply MultilayePerception with before removing attributes in the dataset and after removing the identified attributes from the dataset (veil-type, gill-attachment, and veil-color).

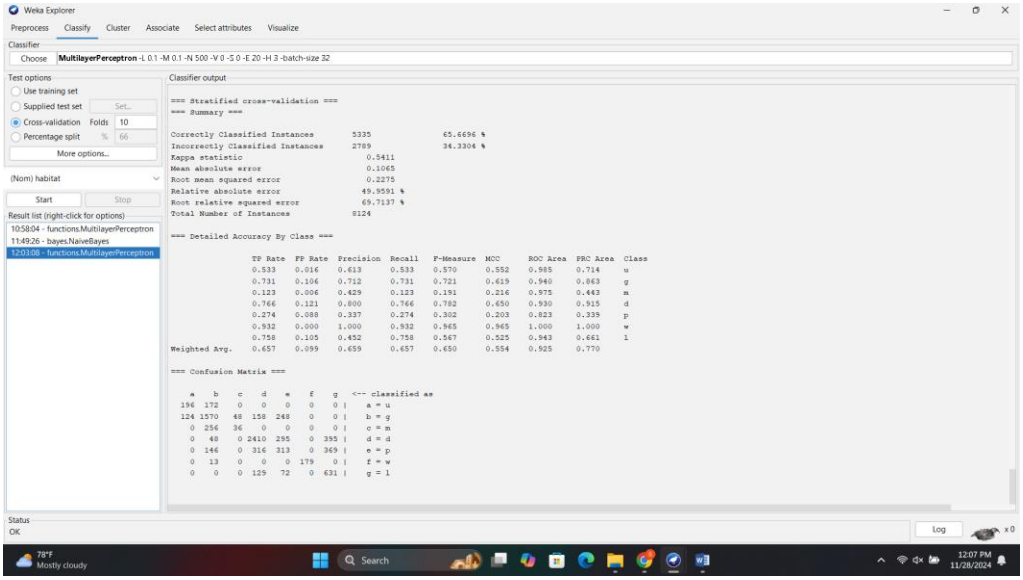

Figure 30: After removing attributes and apply MultilayerPerception

Identified key observation in Accuracy improvement is, correctly classified instances increased from 60.33% to 65.66%.

Apply NaiveBayes with the updated dataset

Here's a comparison between the apply NaiveBayes with before removing attributes in the dataset and after removing the identified attributes from the dataset (veil-type, gill-attachment, and veil-color).

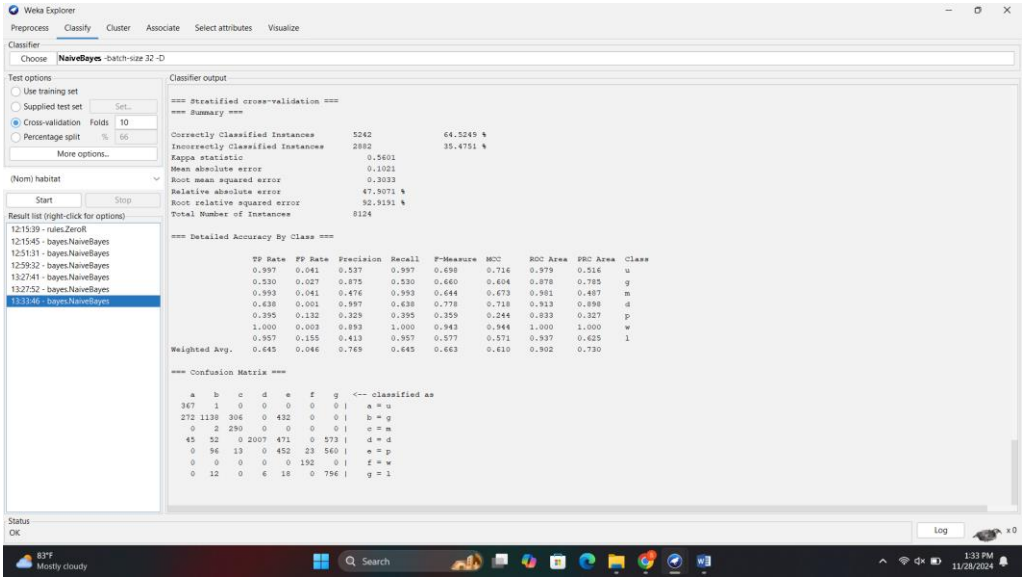

Figure 31: After removing attributes and apply Naive Bayes

4. Justification of the Removal attributes based on the outcome

- Improved Model Performance - Increase accuracy, Kappa statistic and weighted metrics justifies attributes removal.
- Reduced Complexity: Fewer attributes simplify the model and making it more efficient.
- Validation of Attribute Selection: The ranking process (information gain) accurately identified attributes that has low contribution.

The removal of three attributes is justified, because slight improvement in classification performance.

End.