Laboratory 4

Variant 5

Group 5

By Jan Szachno and Aleksandra Głogowska

# 1. Introduction

The task is to write a program that solves a classification task on a given dataset. The goal is to classify the type of wine, based on its properties. The dataset is under this link: https://scikit-learn.org/stable/datasets/toy_dataset.html#wine-dataset. The task requires to analyze the features in the dataset and to decide which ones are useful. It is crucial to use the template that has been provided, with already split data to train and test sets. The next step is to choose two models for classification and explain the choice. The last step before the testing and conclusions is to select an appropriate metric to evaluate the models. The parameters of the models have to be changed, and the performance compared using a cross-validation framework with 4 folds.

# 1. Algorithm Description

The main goal of the program is to classify the type of wine based on its properties. Classification is a fundamental concept in machine learning and statistics. It involves categorizing items into predefined classes or categories based on their features or attributes. The goal of classification is to learn a model from labeled training data that can predict the class labels of unseen or new instances.

Our program and report are based on four tasks, each one consecutively leading to the solution and conclusions of this laboratory. One of them, which is the data split, has already been done for us in the provided Python template.

### *TASK 1*

*Analyze the features in the dataset and decide which ones are useful for your task. Some features might be redundant, for example, a dataset might contain features that depend on each other, such as ”pre-tax price”, ”tax”, and ”post-tax price”, where the third one is just a sum of the previous two. In such cases, it’s better to remove dependant features. If the dataset is not normalized, you should also perform normalization. You can learn more about data preprocessing on sklearn tutorial. Describe your thought process in the report.*

Initially, we decided not to exclude any feature from the dataset without in-depth analysis. Determining which feature is redundant would require specialized knowledge in the wine field. To resolve the problem, in the following steps, we will study the importance of different features based on the results of the correlation matrix as well as another algorithmic method that we will introduce later. Before implementing any changes to the dataset, we normalized it using the “sklearn” library. Normalization of a dataset is the process of rescaling the values of features to a similar scale. This ensures that all features contribute equally to the analysis, preventing variables with larger scales from dominating the model.

To determine which features could be redundant, we wrote a program that checks dependencies between wine features and visualizes them. The visualization is shown in Figure 2.1. The dependency check has been done with a correlation matrix. It checks a correlation between every feature and calculates a correlation coefficient in the range [-1,1]. A correlation coefficient of 1 indicates a perfect positive correlation, meaning that as one variable increases, the other variable also increases proportionally. A correlation coefficient of -1 indicates a perfect negative correlation, where one variable decreases as the other variable increases. Finally, a correlation coefficient 0 indicates no linear relationship between the variables.

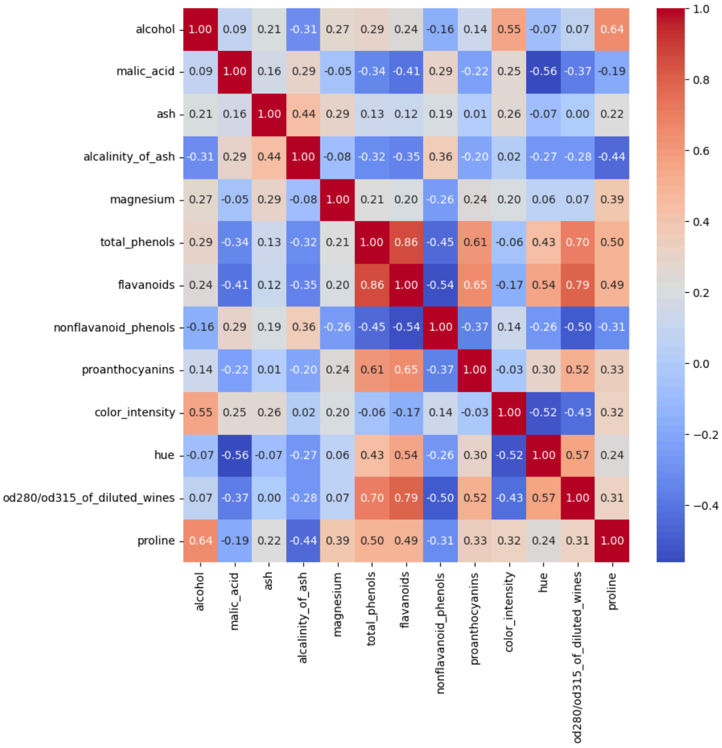

*Figure 2.1. - Visualization of the correlation matrix.*

Next, we filtered the results only to show features correlated with coefficients from 0.79 and higher to find which features are the most dependent on each other. The filtered visualization of the correlation matrix is depicted in Figure 2.2.

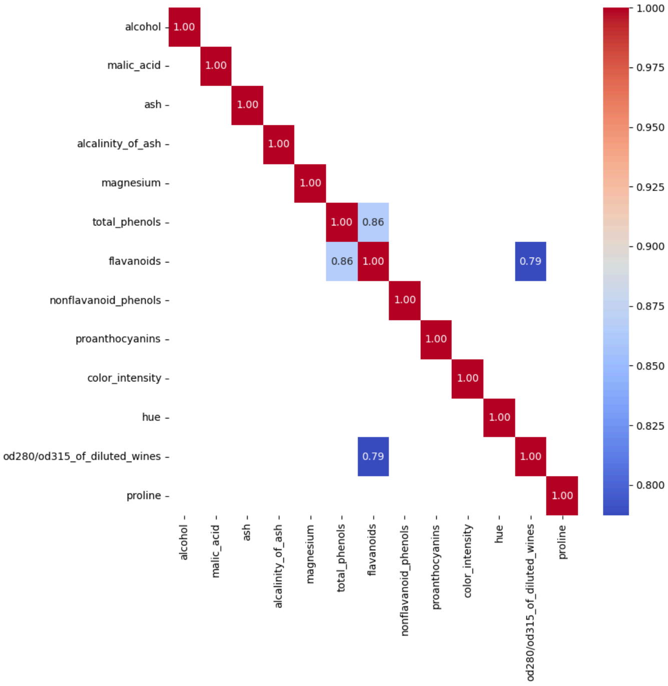

*Figure 2.2. - Filtered visualization of the correlation matrix.*

Based on the correlation matrix analysis, we can see that the highest correlation is between flavanoids and total phenols, as well as flavanoids and od280/od315 of diluted wines. Knowing the correlation of features, we found out which features are potentially excessive. To confirm the presumptions and to understand which features contribute the most to the prediction of our model, we conducted another analysis, which was the computation of feature importance using the Random Forest Classifier method. This technique is widely used for feature importance analysis due to its simplicity and effectiveness. The visualization of the result of this method has been pictured in Figure 2.3.

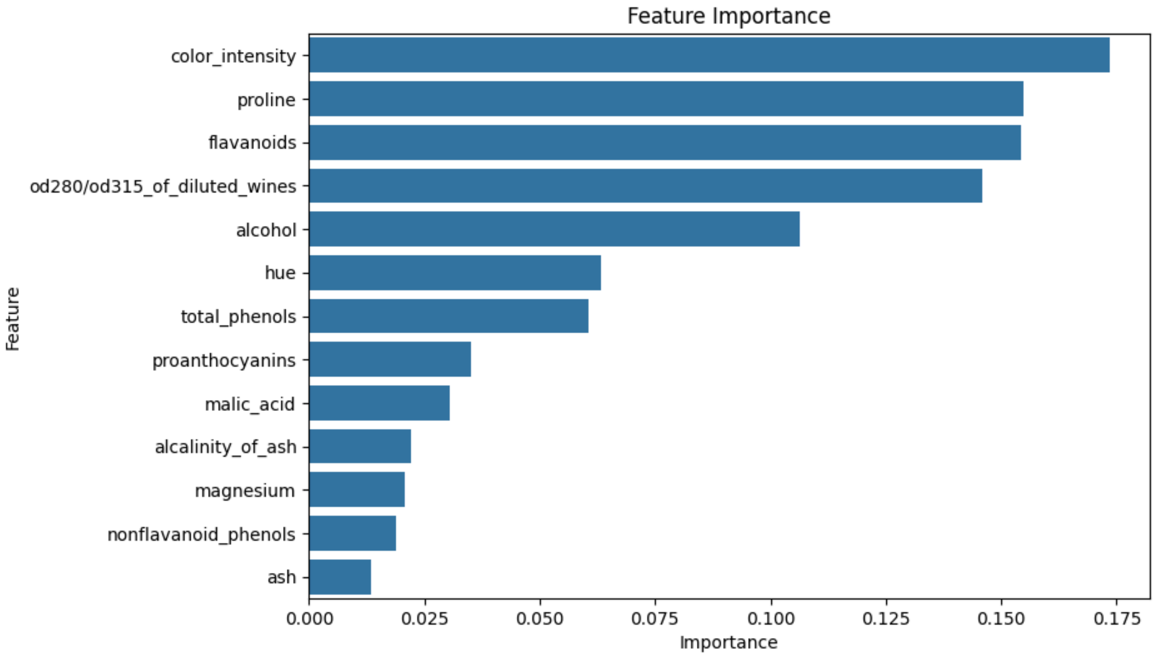

*Figure 2.3. - Visualization of the feature importance using the Random Forest Classifier method.*

Earlier, based on the correlation matrix, we established that flavanoids with total phenols and od280/od315 of diluted wines had the highest correlation. We have to decide which one of them will be excluded from the dataset. The results of the Random Forest Classifier method show that out of these three, flavanoids and od280/od315 of diluted wines have the highest importance. Total phenols display much lower importance, so this feature will no longer be included in the dataset.

## *TASK 2*

*Choose two models to fit, depending on your task, eg.: linear regression, logistic regression, SVM classifier or regressor, decision tree, random forest, etc. You can use any models of your choosing, provided they are suitable to the task (regression or classification). Explain your choice of the models in the report.*

### *Descriptions of each model*

The second task requires choosing an appropriate model to fit depending on the needs of our final goal, which is a classification of the wine types from the dataset. We can choose from models like linear regression, logistic regression, SVM classifier or regressor, decision tree, random forest, etc.

Linear Regression is a statistical method that models the relationship between a dependent variable and one of the more independent variables by fitting a linear equation to observed data. The model equation predicts the dependent variable as a linear combination of the independent variables. It is mostly used for regression tasks. It could be modified so it could be used for binary classification, but it is not optimal and not a common solution.

Logistic Regression is used to model the probability of a binary outcome based on one or more independent variables. It provides the probability that a given input point belongs to a certain class rather than just a binary outcome by using a logistic function to model a binary dependent variable. It is highly suitable and commonly used for binary classification tasks.

SVM (Support Vector Machine) Classifier or Regressor can be used for classification and regression problems. However, it is primarily known for classification. It works by finding the hyperplane that divides a dataset into classes in the feature space. This technique is very suitable for both binary and multi-class classification tasks. It is particularly useful for high-dimensional data and situations where the number of dimensions exceeds the number of samples.

Decision Tree is a structure that is similar to a flowchart. Each internal node represents a test on an attribute, each branch represents the outcome of the test, and each leaf node represents a class label, which is a decision taken after computing all attributes. Trees can be used for both classification and regression. In classification, decision trees are powerful for capturing non-linear relationships and interactions between features without requiring much data transformation.

Random Forrest is an ensemble learning method for classification and regression that works by constructing many decision trees at training time and outputting the class, which is the mode of the classes in case of classification or outputting a mean prediction in regression of the individual trees. This method is very suitable for classification tasks. It works as an improvement over the simplicity of the decision trees by reducing overfitting and improving accuracy through ensemble learning.

In summary, for classification tasks, Logistic Regression, SVM Classifier, Decision Tree, and Random Forest are highly suitable, each having its own strengths in handling different types of data and problem complexities. Linear Regression is generally not preferred for classification unless modified or used under specific conditions.

***Decision on the models choice and explanation***

Knowing the characterizations of each method, we have to choose two models from Logistic Regression, SVM Classifier, Decision Tree, and Random Forest for the classification of three wine types based on 13 features. We chose Random Forest and SVM Classifier.

Random Forest is an ensemble method that can handle both bias and variance effectively, making it robust against overfitting, which is a common problem in complex classification tasks. Overfitting occurs when a machine learning model learns the training data too well, capturing noise or random fluctuations that are not representative of the true relationship between the features and the target variable. This can result in poor performance when the model is applied to new, unseen data, as it fails to generalize well. Random Forest is well suited for datasets with a relatively high number of features, as in the case of the wine classification problem. It has the ability to capture complex interactions between features without the need for feature scaling or extensive preprocessing, which makes this technique a versatile choice. It typically performs well out of the box and can deal with unbalanced data, which is often the case in real-world scenarios.

SVM is powerful for finding the optimal hyperplane that separates the data into classes, which is particularly useful in high-dimensional spaces. This method can capture complex relationships between features and classes, even when the data is not linearly separable. This flexibility allows SVM to achieve high accuracy on a variety of datasets. Furthermore, SVM is effective when the number of dimensions exceeds the number of samples, which can be advantageous in detailed feature-rich datasets.

Random Forest and SVM offer complementary strengths. Random Forest is characterized by its robustness and ease of use, handling both linear and non-linear relationships well through its ensemble of decision trees. SVM, with its flexibility in kernel choice (e.g., linear, polynomial, or radial basis function), excels in capturing complex decision boundaries, especially in high-dimensional spaces. This combination provides a good balance between ease of use, interpretability, and the ability to capture complex patterns in the data.

## 2. Task 3 - Experiments and Conclusions

*Select an appropriate metric to evaluate your models. Try different model parameters for both models and compare their performance using cross-validation framework with 4 folds. Document the 1 parameters you tried for both models in separate tables. Compare the performance of both models with the best parameters you have found.*

The last task is mostly based on testing and conducting experiments on our models. We are going to tweak values of a few parameters to see how they influence the results.

- removal of columns from the dataset - can less data provide better results?
- The scoring parameter - how the model's performance is perceived

K-Fold Cross-Validation is a resampling method commonly used to assess machine learning models on a limited data set. This method has a single parameter, *k*, which denotes the number of groups to split the data into. Typically, the method divides the dataset into *k* equal or approximately equal-sized groups called 'folds'. For each unique group:

- Retain one fold as the validation data for testing the model,
- Use the remaining k−1 folds as training data.

Repeat this process *k* times, with each *k* fold serving as the validation data once. The results from the *k* folds can then be averaged (or otherwise combined) to produce a single estimation. The main benefit of this method is that it uses every observation for both training and validation, and each observation is part of the validation set precisely once. The method evaluates the model's performance across different data subsets, improving the reliability of the model assessment by demonstrating its ability to generalise beyond the training data. The generalisation, however, comes at a cost. An increased computational load arises because the process requires training and validating a model multiple times - once for each fold. This repeated cycle of training and validation across different subsets of the data can significantly boost the computational demands, mainly when the number of folds is large or when the models are complex.
In our experiment, we use cross-validation with 4 folds.

The choice of scoring metric is crucial as it directly affects the feedback on the model's performance. Different metrics will emphasise different aspects of the model's behaviour. The scoring function should ideally reflect the specific priorities and costs associated with different types of errors in the problem domain.

- Accuracy - measures the overall percentage of predictions that are correct. It prioritises overall correctness across all predictions. It is easy to understand and interpret but can be misleading in the presence of imbalanced classes.
- Precision - measures the ratio of correctly predicted positive observations to the total predicted positives. Its priority is preventing false positives. It is advantageous when the cost of a false positive is high. However, it does not consider false negatives, which can be critical depending on the context.
- Recall (Sensitivity) - measures the ratio of correctly predicted positive observations to all observations in the actual class. It prioritises capturing as many positives as possible (preventing false negatives). Useful when the cost of a false negative is high but can lead to a high number of false positives, which can be problematic depending on the application.
- F1 Score - the weighted average of Precision and Recall. It considers both false positives and false negatives. It balances the trade-off between precision and recall. It is beneficial when both false positives and false negatives are costly. However, it is more complex to understand and explain than straightforward metrics like accuracy.

Each metric serves specific priorities in model evaluation and is selected based on what is most critical for the task, such as minimising false positives or maximising the detection of true positives.
Precision might be critical in a spam detection system because false positives (non-spam emails marked as spam) can be more disruptive to users than false negatives (spam emails received in the inbox).

In contrast, recall might be critical in a medical diagnosis application because missing a positive (false negative) diagnosis could be life-threatening. In such cases, the domain knowledge of the doctor would be able to verify the model's predictions.
Choosing the right scoring metric depends not only on the algorithm but also heavily on the application and the cost of different misclassifications.

The following table allows us to observe the results of removing columns from the dataset. We started by removing the total phenols column as we suspected before but it made the results only marginally better and only for random forest model. Later we started removing features which were previously classified as least important. Their removal mainly improved the performance of the random forest classifier and had mostly negative to neutral impact on the support vector machine. In the beginning without removing any features the SVM performed better, however after removing features the random forest achieved better results.

| | SVM cross validation accuracy | Random Forest cross validation accuracy |
|---|---|---|
| No removed columns | 0.979 (+/- 0.023) | 0.972 (+/- 0.020) |
| “Total_phenols” removed | 0.979 (+/- 0.012) | 0.979 (+/- 0.012) |
| “ash” removed | 0.979 (+/- 0.023) | 0.986 (+/- 0.014) |
| “nonflavanoid_phenols” removed | 0.972 (+/- 0.020) | 0.986 (+/- 0.014) |
| “magnesium” removed | 0.979 (+/- 0.023) | 0.979 (+/- 0.012) |
| “alcalinity_of_ash” removed | 0.972 (+/- 0.020) | 0.986 (+/- 0.014) |
| “malic_acid” removed | 0.972 (+/- 0.020) | 0.986 (+/- 0.014) |
| “proanthocyanins” removed | 0.979 (+/- 0.012) | 0.986 (+/- 0.014) |
| Removed all mentioned | 0.979 (+/- 0.023) | 0.986 (+/- 0.014) |

In this experiment we wanted to observe what happens if we remove most important features from the dataset. Surprisingly it turned out that it does not always result in worse results of the models. However we can generally say that removing the features with high importance does result in worse accuracy.

| | SVM cross validation accuracy | Random Forest cross validation accuracy |
|---|---|---|
| No removed columns | 0.979 (+/- 0.023) | 0.972 (+/- 0.020) |
| “color_intensity” removed | 0.965 (+/- 0.023) | 0.951 (+/- 0.023) |
| “proline” removed | 0.986 (+/- 0.014) | 0.986 (+/- 0.014) |
| “flavanoids” removed | 0.972 (+/- 0.020) | 0.979 (+/- 0.012) |
| Removed all mentioned | 0.951 (+/- 0.013) | 0.958 (+/- 0.024) |