CITS5508 Machine Learning
Semester 1, 2024

Assignment 2
Assessed, worth 15%. Due: 8pm, Friday 03th May 2024

Discussion is encouraged, but all work must be done and submitted individually. This assignment has 21 tasks, from which 20 are assessed, which total 45 marks.

You will develop Python code for classification tasks. You will use Grid Search and cross-validation to find the optimal hyperparameters of the model and discuss and interpret the different decisions and their impact on the model's performance and interpretability.

1 Submission

Your submission consists of **two files**. The first file is a report describing your analysis/results. Your analysis should provide the requested plots, tables and your reflections about the results. Each deliverable task is indicated as **D** and a number. Your report should be submitted as a ".PDF" file. Name your file as `assig1_<student_id>.pdf` (where you should replace `<student_id>` with your student ID).

The second file is your Python notebook with the code supporting your analysis/results. Your code should be submitted as `assig1_<student_id>.ipynb`, the Jupyter notebook extension.

Submit your files to LMS before the due date and time. You can submit them multiple times. Only the latest version will be marked. Your submission will follow the rules provided in LMS.

Important:

- You must submit the first part of your assignment as an electronic file in PDF format (do not send DOCX, ZIP or any other file format). Only PDF format is accepted, and any other file format will receive a zero mark.
- **You should provide comments on your code**.
- You must deliver parts one and two to have your assignment assessed. That is, your submission should contain your analysis and your Jupyter notebook with all coding, both with appropriate formatting.
- By submitting your assignment, you acknowledge you have read all instructions provided in this document and LMS.
- There is a general FAQ section and a section in your LMS, Assignments - Assignment 2 - Updates, where you will find updates or clarifications about the tasks when necessary. It is your responsibility to check this page regularly.

- You will be assessed on your thinking and process, not only on your results. A perfect performance without demonstrating understanding what you have done won't provide you marks.
- **Your answer must be concise. A few sentences (2-5) should be enough to answer most of the open questions. You will be graded on thoughtfulness.** If you are writing long answers, rethink what you are doing. Probably, it is the wrong path.
- You can ask in the lab or during consultation if you need clarification about the assignment questions.
- You should be aware that some algorithms can take a while to run. A good approach to improving their speed in Python is to use the vectorised forms discussed in class. In this case, it is strongly recommended that you start your assignment soon to accommodate the computational time.
- For the functions and tasks that require a random procedure (e.g. splitting the data into 80% training and 20% validation set), you should set the seed of the random generator to the value "5508" or the one(s) specified in the question.

2 Dataset

In this assignment, you are asked to train a few decision tree classifiers on the *Breast cancer wisconsin (diagnostic) dataset* available on Scikit-Learn and compare their performances.

Description about this dataset can be found on the Scikit-Learn web page:

https://scikit-learn.org/stable/datasets/toy_dataset.html#breast-cancer-wisconsin-diagnostic-dataset

There are two classes in the dataset:

- *malignant* (212 instances, class value 0) and
- *benign* (357 instances, class value 1).

Follow the example code given on the web page

https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_breast_cancer.html#sklearn.datasets.load_breast_cancer

to read the dataset and separate it into a feature matrix and a class vector. Your feature matrix should have 569 (rows) $\times$ 30 (columns) and your class vector should have 569 elements.

In all asked implementations using Decision Trees, Random Forests or data splits (such as when using `train_test_split()`), you should set `random_state` as specified for results reproducibility. You should aim to round your results to the second decimal place.

3 Tasks

First inspections on the dataset and preprocessing

D1 **2 marks**

Re-order the columns in your feature matrix (or dataframe) based on the column name. Provide a scatter plot to inspect the relationship between the first 10 features in the dataset. Use different colours in the visualisation to show instances coming from each class. (Hint: use a grid plot.)

D2 **2 marks**

Provide a few comments about what you can observe from the scatter plot:

- What can be observed regarding the relationship between these features?
- Can you observe the presence of clusters of groups? How do they relate to the target variable?
- Are there any instances that could be outliers?
- Are there features that could be removed? Why or why not?

D3 **1 mark**

Compute and show the correlation matrix where each cell contains the correlation coefficient between the corresponding pair of features (Hint: you may use the `heatmap` function from the `seaborn` package).

D4 **1 mark**

Do the correlation coefficients support your previous observations?

D5

In a data science project, it's crucial not just to remove highly correlated features but to consider the context and implications of feature selection carefully. Blindly dropping features may lead to the loss of valuable information or unintended bias in the model's performance. Here, for the assignment context, we will drop a few features to simplify the classification tasks and speed up the computational time. Create a code that drop the features: `mean perimeter`, `mean radius`, `worst radius`, `worst perimeter` and `radius error`. These are features with a linear correlation higher than `0.97` in magnitude with some other features kept in the data.

After this process, your data matrix should be updated accordingly and contain 25 features. Task **D5** must be performed; otherwise, your order deliverable tasks will be incorrect. However, there are no marks for task **D5**.

Fitting a Decision Tree model with default hyperparameters

D6 **3 marks**

Fit a decision tree classifier using default hyperparameters using 80% of the data. Remember to set the random generator's state to the value "5508" (for both the split and class). Use the trained classifier to perform predictions on the training and test sets. Provide the accuracy, precision and recall scores for both sets and the confusion matrix for the test set.

D7 **2 marks**

Comment on these results. Do you think your classifier is overfitting? If so, why this is happening? If not, why not?

D8 **2 marks**

Display the decision tree built from the training process (like the one shown in Figure 6.1 of the textbook for the iris dataset).

D9 **2 marks**

Study the tree diagram and comment on the following:

- How many levels resulted from the model?
- Did the diagram help you to confirm whether the classifier has an overfitting issue?
- What can you observe from the leaves?
- Is this an interpretable model?

D10 **3 marks**

Repeat the data split another four times, each using 80% of the data to train the model and the remaining 20% for testing. For these splits, set the seed of the random state to the values "5509", "5510", "5511" and "5512". The random state of the model can be kept at "5508".

For each of these four splits, fit the decision tree classifier and use the trained classifier to perform predictions on the test set. Provide three plots to show the accuracy, precision and recall scores for the test set for each split and comment on the consistency of the results in the five splits (including the original split with random state "5508").

D11 **3 marks**

Investigate the impact of the training size on the performance of the model. You will do five different splits: 50%-50% (training the model on 50% of the data and testing on the remaining 50%), 60%-40%, 70%-30%, 80%-20% and 90%-10%. For each of these data splits, set back the seed of the random state to the value "5508".

Provide three plots to show the accuracy, precision and recall scores for the test set for each data split and comment on the results. Did the performance behave as you expected?

Fitting a Decision Tree model with optimal hyperparameters

D12 **4 marks**

Create a training set using 80% of the data and a test set with the remaining 20%. Use a 10-fold cross-validation and grid-search to find the optimal combination of hyperparameters `max_depth` — using values `[2, 3, 4, 5]`, `min_samples_split` — using values `[2, 4, 5, 10]`, and `min_samples_leaf` — using values `[2, 5]` of a decision tree model. Remember to set the seed of the random state of the data split function and model class to the value "5508". For the cross-validation, set the value of the random state to "42". Use `accuracy` for the `scoring` argument of the grid-search function.

With the optimal obtained hyperparameters, retrain the model and report:

- The optimal hyperparameters;
- The obtained accuracy, precision and recall on the training set;
- The obtained accuracy, precision and recall on the test set;
- The confusion matrix on the test set.

D13 **2 marks**

Comment: What was the impact of fine-tuning the hyperparameters as opposed to what you obtained in **D6**? Has fine-tuning done what you expected?

D14 **3 marks**

Repeat the training of task **D12** twice: one considering the `scoring` argument of the grid-search function as `precision` and the other `recall`.

For each of the scoring options (accuracy, precision, recall), provide the optimal hyperparameters according to the 10-fold cross-validation and grid-search, and, after retraining each model accordingly, provide the confusion matrix on the test set. Comment on the results, considering the problem.

Fitting a Decision Tree with optimal hyperparameters and a reduced feature set

D15 **1 mark**

Using the model with fine-tuned hyperparameters based on `accuracy` (the one you obtained in **D12**), display the *feature importance* for each feature obtained from the training process. You should sort the feature importances in descending order.

D16 **3 marks**

Using the feature importance you calculated in the previous task, trim the feature dimension of the data. That is, you should retain only those features whose *importance* values are above 1% (i.e., 0.01). You can either write your own Python code or use the function `SelectFromModel` from the `sklearn.feature_selection` package to work out which feature(s) can be removed.

Report what features were retained and removed in the above process. Also report the total feature importance value that is retained after your dimension reduction step.

D17 **3 marks**

Compare the model's performance (accuracy, precision, recall) on training and test sets when using the reduced set of features and the model trained on the complete set of features. Also, report the corresponding confusion matrices on the test sets. (You will need to consider whether you should repeat the cross-validation process to find the optimal hyperparameters).

D18 **1 mark**

Comment on your results. What was the impact (if any) of reducing the number of features?

Fitting a Random Forest

D19 **3 marks**

Considering all features and the 80%-20% data split you did before, use 10-fold cross-validation and grid-search to find a Random Forest classifier's optimal hyperparameters `n_estimators` (number of estimators) and `max_depth`. Remember to set the seed of the random state of the data split function and model class to the value "5508". Use `n_estimators:[10, 20, 50, 100, 1000]`, and `max_depth:[2, 3, 4, 5]`. For the cross-validation, set the value of the random state to "42". Use `accuracy` for the `scoring` argument of the grid-search function.

Keep the other hyperparameter values to their default values. Use the optimal values for the `n_estimators` and `max_depth` hyperparameters to retrain the model and report:

- The obtained optimal number of estimators and max depth;
- The obtained accuracy, precision and recall on the training set;
- The obtained accuracy, precision and recall on the test set;
- The confusion matrix on the test set.

D20 **2 marks**

How do these performances compare with the ones you obtained in **D12**? What changed with the use of a Random Forest model? Is this result what you would expect?

D21 **2 marks**

Thinking about the application and the different models you created, discuss:

- Do you think these models are good enough and can be trusted to be used for real?
- Do you think a more complex model is necessary?
- Do you think using a machine learning algorithm for this task is a good idea? That is, should this decision process be automated? Justify.
- Are there considerations with the used dataset?