MEM T380 – Applied Machine Learning in Mechanical Engineering

Case Studies Assignment 2

Part C - Ensemble Learners & Comparison of Classifiers

Dimitrios Fafalis, PhD

Dimitrios.Fafalis@drexel.edu

due Thursday, May 18, 2023

Students' names & ID:

1. ______________________________ ____________

2. ______________________________ ____________

Submit your files (.ipynb) and a report (.pdf) on Blackboard by due date.

Classifying Weld Defects!

CASE STUDY 1. points 100 – **Weld defects classification**

This case study is related to classifying weld defects and is based on the research article "**Lu Yang and Hongquan Jiang, 2020**, *Weld defect classification in radiographic images using unified deep neural network with multi-level features*, Journal of Intelligent Manufacturing ". This article is available in the Case Studies folder for HW-2. Although the authors of the paper used an artificial neural network (ANN) machine learning approach to classify the weld defects, in this and the following assignments you will use the k-Nearest Neighbors and the Decision Trees classification techniques. In a later lecture, we will revisit this case study and practice with ANN. Refer to this article to familiarize yourselves with the problem.

The raw data for this case study are available to you in an excel workbook posted in the assignment `weld_defect_dataset.xlsx`. The data are partitioned into five subsets, each one stored in a separate sheet named as `subset1`, ..., `subset5`.

The data come from image processing of radiographic images. The features and the target classes have been extracted from those images. The data consist of eleven (11) total features and seven (7) weld defect types. How the features were extracted from the radiographic images is a completely different topic. You are welcome to reach out to me to guide you into literature and tools on how to do that.

The goal of this case study is to develop various ML models that classify weld defects observed in radiographic images.

1 Data Exploration Tasks

0 points: this task has been performed in HW2/Part A; reuse it here as a prerequisite for the new tasks. The very first steps in developing a `Machine Learning` model are to load, explore, and preprocess the data. The goal of this task is to explore the data and try to listen to what they want to tell us!

1. using the `pandas` command `read_excel`, load the datasets available into the five sheets of the file `weld_defect_dataset.xlsx` and store them into `pandas` dataframes. It is advised to create a different dataframe for each subset so that you can do cross-validation when you explore the classification models. Refer to pandas.read_excel documentation for examples and additional options.

2. print a summary of the information included in the dataframes using the functions `df.info`, `df.dtypes` and `df.describe`.

3. identify whether there are missing values in any of the fields (columns) of the dataframes.

4. identify whether there are duplicated entries in the dataframes.

5. if there are missing values, return a part of the table that contains only the entries (rows) of the missing data (include all columns). Do you think that the missing data in the particular fields (columns) would have an important impact on interpreting the data of the other fields? Create a new dataframe that contains only non-missing data.

6. if the original dataframes had missing or duplicated entries, print the properties of the *cleaned* dataframe, using the commands `df.info` and/or `df.describe` .

7. identify which of the columns (features) could be used as `categorical` data. These could be `cell` or `string` arrays or even `numeric` data with distinct and repetitive values along all entries. Convert these columns into `categorical` type.

8. use the `seaborn` command `pairplot` to visualize `bivariate` relationships between the `numerical` fields grouping them by the `categorical` fields. Repeat this task as many times as the number of categorical features of the original dataframe (*Hint*: in our case only the target classes is categorical).

9. create a heatmap of the correlation matrix of the features.

10. identify from the figures in item 8 which `numerical` features (fields) are the **strongest** to be used in classifying the data in any of the groups in the `categorical` fields; in other words, which features distinguish the data in groups as clear as possible. Do you see any patterns or trends?

11. are your insights from item 10 justified by looking at the heatmap correlation matrix in item 9?

12. create any other figures using `matplotlib` or `seaborn` that would help you better understand your data. E.g., are they imbalanced? what is the range of values for the features, are there significant differences, etc.

2 Classification with Ensemble Algorithms

40 points. For the following tasks combine all subsets into one large dataset. Also, use **all** features and **all** target defect classes.

1. Combine all five subsets into one. You may look at the `pandas` documentation Merge, join, concatenate and compare.

2. Split the combined dataset into a training set and a testing set, using a 20% test size. Use the method `train_test_split` of the `sklearn.model_selection` package.

3. Create the following four models using the ensemble algorithms available in the `sklearn` package, and the **training** dataset:

   - a **Random Forest** classifier, `RF`, using the `sklearn.tree.RandomForestClassifier` class, and the **training** dataset. Experiment with different values of `n_estimators`, `max_depth`, and `criterion`. What do you observe?
   - a **Bagging** classifier, `BG`, using the `sklearn.ensemble.BaggingClassifier` class. Experiment with different values of `n_estimators`. What do you observe?
   - an **AdaBoost** classifier, `AB`, using the `sklearn.ensemble.AdaBoostClassifier` class. Experiment with different values of `n_estimators` and `learning_rate`. What do you observe?
   - a **Gradient Boosting** classifier, `GBC`, using the `GradientBoostingClassifier` class of the `sklearn.ensembler` package. Experiment with different values of `n_estimators` and `learning_rate`. What do you observe?

4. For each of the models you created in item 3, use the `predict` method on the testing dataset and report the `accuracy_score` from the `sklearn.metrics` library. Collect your results into a nice dataframe, in a similar way it was shown in class.

5. For each of the models you created in item 3, calculate the confusion matrix using the `sklearn.metrics.confusion_matrix` command of the `sklearn` package and display it using the `seaborn` command `heatmap`.

6. For each of the models you created in item 3, provide the classification report using `classification_report` function of the metrics library of the `sklearn` package. Which model performs the best and why?

3 Random Forests vs Decision Trees Classifiers

20 points. For the following tasks combine all subsets into one large dataset. Also, use **all** features and **all** target defect classes. In this section you are requested to compare a random forest with two simple decision trees. Additionally, you will determine the optimal number of trees in the forest for optimal performance in terms of accuracy, using a cross validation technique.

1. Create two simple decision trees, one with `max_depth=1` and one with `max_depth=3`, using the class `sklearn.tree.DecisionTreeClassifier` of the `sklearn` package.

2. Create a random forest classifier using the `sklearn.tree.RandomForestClassifier` class. Let the number of trees, a.k.a. `n_estimators`, vary from 1 to 100, while fixing the `max_depth=3`. That is, you will create 100 random forest models with increasing number of decision trees.

3. Use the `cross_val_score` function from the `sklearn.metrics` library to calculate the accuracy of the models above for a maximum of ten folds, a.k.a. `cv=10`.

4. Plot in a common figure the accuracy performance of the models above as a function of the number of trees in the random forest.

5. What do you observe? Does the random forest model reach a high accuracy level? At which optimal number of trees?

Hint 1: You may wish to consult §12.5 of your textbook by [Fenner, 2020], on evaluating and comparing various classifiers.

Hint 2: Alternatively, you may wish to use the function `validation_curve` from the `sklearn.model_selection` library, as shown in W5 lectures.

4 Evaluating Classifiers

40 points. For the tasks of this section combine all subsets into one large dataset. Also, use **all** features and **all** target defect classes.

1. For the tasks of this section you will need the settings of the models you created in item 3 of section 2 of the current homework, **and** the optimal `k-NN` model you developed in Part 2 (section 2.2) of Homework Assignment 2/Part A.
2. Create the optimal `k-NN` model from homework assignment 2/A.
3. Use $K = 10$ folds for the cross validation techniques required in the following tasks
4. For each of the five target classes of the problem, create a common `ROC` curve (Receiver Operating Characteristic curve) for the models in item 1 and 2 above. Use the functions `cross_val_predict` from the `sklearn.model_selection` library and the `roc_curve` method of the `sklearn.metrics` library to retrieve cross validation probabilities, the false positive rate and the true positive rate.
5. Additionally, calculate the area under curve metric for the cases above using the method `roc_auc_score` from the `sklearn.metrics` library. This metric should be displayed in the legend of the `ROC` curve you created above.
6. Finally, calculate the cross-validation scores for each model, using the `cross_val_score` method of the `sklearn.model_selection` library. This metric should be displayed in the legend of the `ROC` curve you created above.
7. Observing the `ROC` curve for each weld defect, which method do you think is optimal? Comment thoroughly.

Hint: You may wish to consult §6.7.1 of your textbook by [Fenner, 2020], on evaluating and comparing various classifiers.

References

[Fenner, 2020] Fenner, M. (2020). *Machine Learning with Python for Everyone.* Addison-Wesley Professional, -, 1st edition.