Week 8
Performance metrics for classification, Balance and Scaling issues

Dataset: Wine Quality

The attached zip file contains:

- winequality.names : a text file that describes the dataset, including all attributes.
- winequality-red.csv : red-wine dataset (1599 wines)
- winequality-white.csv : white-wine dataset (4898 wines)

Every wine was graded with quality score by experts.

Task:

Develop a classification model for classifying the quality of an unseen wine, from the measured set of attributes, as good or bad. The threshold score for good wine is 7 i.e. score of 7 or higher is good.

1) Learn (via online-search and discussion) the meaning of the following terms.
   - True positives (TP)
   - False positives (FP)
   - True negatives (TN)
   - False negatives (FN)

2) Interpret the definition of each the following performance metrics and discuss its implication.

   - Accuracy = $\frac{TP+TN}{TP+FP+TN+FN}$

   - Precision = $\frac{TP}{TP+FP}$

   - Recall (or Sensitivity) = $\frac{TP}{TP+FN}$

     - the opposite of Recall, with focus on negatives, is called Specificity

   - F1 Score = $\frac{2}{\frac{1}{\text{Precision}}+\frac{1}{\text{Recall}}}$ (the harmonic mean of Precision and Recall)

3) Decide on how you will approach the problem, for example;
    - Red wines only
    - White wines only
    - Combined red and white wines
        - For this option, decide whether the wine type is required as an additional attribute (feature) ?

4) Split the dataset into training set and testing set. For those who understand the usage of validation set, the testing set may be used as the validation set in this learning session. Use any method that does not take too much for this step.
    - An additional column, the quality classification (as good or bad) should be added, so that extracting the target output for the model is convenient.

5) Code a notebook to create a neural network model for classification. There will be a number of options to choose for the model, for example,
    - The number of hidden layers
    - The number of neurons in each hidden layers
    - The activation functions
    - The Keras model to be used (Model or Sequential)
        - I did not see the need to cover Sequential model in class. If you want to use it, study how via online-search. It is virtual simple to understand.
    - The optimizer (I recommend using Adam, however)
    - The number of epochs and the batch size.
    - Use of validation set
        - You may proceed if you understand why and how, or I will bring this into discussion after the model has been created

6) Code a cell to calculate performance metrics on the testing set results; accuracy, precision, recall, and f1. For this problem, it is natural to assume that “good” is positive.

7) Tune the model by iteratively adjusting the variables involved in the model development. This may also include the management of training and test sets.
    - Explore the scaling of data into common value range. This needs additional coding on treating the input data.
        - Do you find scaling help improving/worsening the learning performance?

8) *Summarize in a notebook text cell* about what you eventually make use of the performance metrics. What are the approaches that you found to help optimize the values of performance metrics altogether.

Appendix

A useful data analysis tool is the “Pearson correlation”. The following code gives a quick visualization of correlation between every attribute (X) and the target variable (y). Accordingly, you may decide on selecting only a subset of features to create the model.

```
import numpy as np
import matplotlib.pyplot as plt
from yellowbrick.target.feature_correlation import feature_correlation

visualizer = feature_correlation(X, y)
plt.tight_layout()
```