Lab 3: Comparing Classifiers

This lab is due at 11:59 PM ET on Sunday, October 20, 2019. Late submissions will be penalized as described in the syllabus.

You may work individually or in groups of two. If working in a group, only one student needs to submit the group's work via Carmen. This lab is worth 125 points.

As with previous labs, all your coding should be done in one or more Jupyter notebooks, which you will submit. However, the notebook will not be graded. ALL of your observations, results, and analysis must be provided in report form. **Only the report will be graded.** If something is missing from the report, you will not get credit for it, even if it appears in your notebook. The report must be typewritten (preferably in Microsoft Word or PDF format) and must include the following:

- The usual metadata: title, name(s), date, etc. Make sure both names are included, if working in a group of two.
- Any visualizations relevant to your analysis, with captions or labels.
- All information specifically required in the following sections.

In this lab, you'll use the Scikit-Learn[1] library, one of numerous SciKits (SciPy toolkits) for scientific computation in Python. You'll use several different Scikit-Learn classifiers to try to predict the grape variety of wines based on 13 feature attributes.

0. Get the data

Download the "Wine" dataset from the UCI Machine Learning Repository[2]. (Note this is not the same as "Wine Quality" dataset.) The dataset contains 178 rows and 14 columns.

1. Exploratory data analysis (15 points)

Discuss any observed patterns or data quality issues, such outliers, missing values, and attribute correlations. Visualize those that are interesting or important. Include a dataset description and summary statistics, as always. Describe your observations.

2. Data preprocessing (10 points)

Determine whether any transformations or data preprocessing (e.g. feature subset selection, scaling, feature creation) are needed. Apply the transformations (or don't). Describe and justify your decisions—i.e. what did you do and why? Be sure to include data descriptions (as above) for any new/engineered features.

Split your data into training and validation sets using the **`sklearn.model_selection.train_test_split`** splitter function and **`test_size=0.5`**. Set the test set aside and do not use it again until part 5.

[1] Scikit-Learn API reference: scikit-learn.org/stable/modules/classes.html
[2] UCI ML Repo: archive.ics.uci.edu/ml

3. Build and evaluate six classifiers (50 points)

In this part, you will build the six classifiers listed below. In all cases, use the default hyperparameters.

- Logistic Regressions **`(sklearn.linear_model.LogisticRegression)`**
- k-Nearest Neighbors **`(sklearn.neighbors.KNeighborsClassifier)`**
- Decision Tree **`(tree.DecisionTreeClassifier)`**
- Naïve Bayes **`(sklearn.naive_bayes.GaussianNB)`**
- Artificial Neural Network **`(sklearn.neural_network.MLPClassifier)`**
- Support Vector Machine **`(svm.LinearSVC)`**
- Ensemble Classifier (**`ensemble.RandomForestClassifier)`**

For each classifier, you must:

- Fit the model using the training set.
- Make predictions on the training set using the **`predict()`** method.
- Report performance metrics for the training set—accuracy, F-measure, plus any others you think are interesting.

Be sure to use ONLY the training set for this part!

Based on the results so far, what is your preferred classifier? Justify your choice by comparing results to the other classifiers. What are the pros and cons of your preferred classifier? Report all performance metrics, as well as any other interesting observations.

4. Make predictions and evaluate performance on the test set (10 points)

For this part, you'll return to the test set that you set aside in part 2. Make predictions on the test set using the models you built in part 3. Compare the performance metrics for the test set with those from the training set. Which models appeared to generalize well? For those that did not, provide a possible explanation.

5. Improving your models (30 points)

For this part, choose ONLY ONE of the following items to include in your analysis:

- Without re-splitting the original data set, use a hyperparameter optimizer (e.g. **`model_selection.GridSearchCV`**) on the training set to determine optimal parameters for one of the six models you built in part 3. Use the optimal model to rerun predictions on the test set. Report your findings.
- Choose an alternative sampling approach to split the original dataset into training and test sets. (Refer to the **`model_selection`** module of Scikit-Learn to see what options are available.) Using only exploratory data analysis techniques (including any visualizations you find compelling), compare the training and test sets produced in part 2 with those produced using your "better" technique. Explain your findings.
- Choose one of the models you built in part 3 and build a ROC curve using the training set. Assuming that Scikit-Learn always uses a threshold value of $t = 0.5$, can you identify a better threshold value? Report your findings.

6. Conclusions (10 points)

Summarize your findings and provide any **observations, insights, or opinions** you have. Specifically, which model(s) performed the best? What, if any effect did the work you did in part 5 have on model performance. **Support your assertions** with evidence (facts from your analysis) wherever appropriate. Include a description of any challenges you had or of future steps you might take to further improve upon your results.

7. Submit your work

Submit your notebook(s) and your typewritten report via Carmen, under “Lab 3”. Do not submit any data files.