CS578 HW1

September 5, 2018

Total Points: 100 (Max possible 115)
Due date: 11:59pm Wednesday Sptember 19, 2018.

1 Task

1.1 Objective

The objective of this assignment is to experiment on Decision Tree classifier and KNN classifier. You will predict the quality of white wine in a scale of 0 to 10 using these two classification algorithms and compare the results. You need to implement this assignment from scratch in Python 2.7. Do not use any already implemented models like those in the scikit-learn library.

1.2 Dataset

The dataset is contained in `winequality-white.csv`[1]. There are 4898 instances and 11 features associated with each instance. The description of the features (presented in corresponding columns in `winequality-white.csv`) are as follow.

```
column 1 - fixed acidity
column 2 - volatile acidity
column 3 - citric acid
column 4 - residual sugar
column 5 - chlorides
column 6 - free sulfur dioxide
column 7 - total sulfur dioxide
column 8 - density
column 9 - pH
column 10 - sulphates
column 11 - alcohol
All the features are continuous valued.
```

[1] https://archive.ics.uci.edu/ml/datasets/Wine+Quality

The Label of each data point is represented in `column 12` (You need to predict this!!!).

```
column 12 - quality (score between 0 and 10)
This field is integer valued.
```

1.3 What To Do?

- Do 4-fold cross validation i.e. split the whole dataset into 4 folds and use one fold as test and other three combined as training set each time. Tune various hyper-parameters using validation set i.e. each time keep some data points aside from the training set and tune the hyper-parameters while training based on the performance on validation set. **Clearly put a pointer to the portion of code which does this task in report and comment adequately on the script also for points. (10 points)**
- Implement a Decision Tree using ID3 algorithm. Tune the max-depth using validation set and report the best performing one. **(5 points if the code executes without error + 25 for correctness of the implementation)**
- Implement KNN classifier. Use at least three distance measures and report the best performing one. (Should take the decision based on the performance on the validation set) Tune the value of K and report the best performing one. **(5 points if the code excutes without error + 20 for correctness of the implementation)**

1.4 Scope of Creativity

Possibly you need to categorize the continuous valued features for Decision Tree. There is a scope of being creative here. One obvious thing you may try to convert the continuous valued features to categorical is to set thresholds on the values. There may be unknown values in the features of the test set which are not seen while training. How will you deal with this problem? You may normalize the continuous values by using sigmoid or any other function you prefer. We encourage you to experiment on this. **Clearly mention in your report how did you handle these issues and do the categorization. (5 points)**

2 Report

- The performance metrics will be average F1 score and average accuracy over all the folds.
- Report training, validation and test accuracy and F1 scores for all of the folds for both Decision Tree and KNN in tabular format. There will be 48 data points for 4-folds. For example, training, validation and test accuracy and F1 scores for each fold i.e. 3*2=6 data points for each fold and there are 4 folds, so in total 6*4=24 data points for Decision Tree and another 24 for KNN. **(5 points)**

- Show validation accuracy and F1 score against max-depth for Decision Tree in graphical format. (You can use gnuplot for generating graphs) **(5 points)**
- Show validation accuracy and F1 score against distance measures and K for KNN in graphical format (2 graphs). **(10 points)**
- Answer the following questions briefly (not more than 3 lines each) in your report. **Clearly mention the question number beside every answer.**
  1. What is most likely to happen if you allow the max-depth upto the number of features in a Decision Tree? **(3 points)**
  2. What are the basic differences between a Decision Tree classifier and a KNN classifier? **(3 points)**
  3. How would you convert your decision tree (in the depth and prune cases) from a classification model to a ranking model? **(4 points)**

3 Bonus Points

Create a separate section in your report and name it **"Additional Works"** and precisely mention what you have done for bonus points. **Clearly mention only the results of your best performing model in this section again. (If you have done nothing extra, still mention your best result again here with the model and it's parameters which yields this best result. This is necessary for the bonus points)** The following are the scopes of getting bonus points.

- Experiment on Post Pruning on Decision Tree for bonus points. Clearly indicate the effects of this in your model using the same format in Section 2 **(5 points max)**
- Report your best result here and top 15 best scores get bonus **5 points**.
- Anything else which helped to improve the models. **(5 points max)**

4 Submission

4.1 Mandatory Files

- Your submission folder should contain 3 files:
  1. A file named **`decisiontree.py`**: This script should contain the full implementation of Decision Tree classifier. It should not take any argument from the command line. Upon running this script the output should be the best result (F1 score and accuracy) of Decision Tree classifier which you are reporting in your report. The result should be of training, test and validation set of all the folds and the average one over all of the folds. Please strictly follow the following input/output format, otherwise we will penalize some points.

```
$ python decisiontree.py
```

```
Hyper-parameters:
Max-Depth: 6

Fold-1:
Training: F1 Score: 86.7, Accuracy: 88.9
Validation: F1 Score: 86.7, Accuracy: 88.9
Test: F1 Score: 86.7, Accuracy: 88.9

Fold-2:
Training: F1 Score: 84.1, Accuracy: 86.4
Validation: F1 Score: 84.1, Accuracy: 86.4
Test: F1 Score: 84.1, Accuracy: 86.4

Fold-3:
Training: F1 Score: 81.7, Accuracy: 83.5
Validation: F1 Score: 81.7, Accuracy: 83.5
Test: F1 Score: 81.7, Accuracy: 83.5

Fold-4:
Training: F1 Score: 89.7, Accuracy: 90.5
Validation: F1 Score: 89.7, Accuracy: 90.5
Test: F1 Score: 89.7, Accuracy: 90.5

Average:
Training: F1 Score: 85.6, Accuracy: 87.3
Validation: F1 Score: 85.6, Accuracy: 87.3
Test: F1 Score: 85.6, Accuracy: 87.3
```

2. A file named **knn.py**: This script should contain the full implementation of KNN classifier. It should not take any argument from the command line. Upon running this script the output should be the best result (F1 score and accuracy) of KNN classifier which you are reporting in your report. The result should be of training, test and validation set of all the folds and the average one over all of the folds. Please follow the following input/output format, otherwise we will penalize some points.

```
$ python knn.py
Hyper-parameters:
K: 9
Distance measure: Cosine Similarity

Fold-1:
Training: F1 Score: 86.7, Accuracy: 88.9
Validation: F1 Score: 86.7, Accuracy: 88.9
```

```
Test: F1 Score: 86.7, Accuracy: 88.9

Fold-2:
Training: F1 Score: 84.1, Accuracy: 86.4
Validation: F1 Score: 84.1, Accuracy: 86.4
Test: F1 Score: 84.1, Accuracy: 86.4

Fold-3:
Training: F1 Score: 81.7, Accuracy: 83.5
Validation: F1 Score: 81.7, Accuracy: 83.5
Test: F1 Score: 81.7, Accuracy: 83.5

Fold-4:
Training: F1 Score: 89.7, Accuracy: 90.5
Validation: F1 Score: 89.7, Accuracy: 90.5
Test: F1 Score: 89.7, Accuracy: 90.5

Average:
Training: F1 Score: 85.6, Accuracy: 87.3
Validation: F1 Score: 85.6, Accuracy: 87.3
Test: F1 Score: 85.6, Accuracy: 87.3
```

3. A written report using LaTeX named **`report.pdf`**. Your report should contain your name and Purdue e-mail id.
4. **(optional)** A `README` file containing your name, instructions to run your code and anything you would like us to know about your program (like errors, special conditions, additional commands for the bonus points etc).
5. The bound on runtime is 5 minutes for each script. If your scripts take more than that, mention that in the `README` file. If your scripts take excessively more time to execute than what is required in general (for most of the other students), some points will be penalized for that.

4.2 Submission Procedure

You are required to use LaTeX to type your solutions to questions, and report of your programming as well. https://www.overleaf.com/ is a website you can use freely as a Purdue student. Other formats of submission will **not** be accepted. A template named "homework_template.tex" is also provided for your convenience.

Your code will be tested on `data.cs.purdue.edu`, where you submit your homework as well. **Make sure that your program runs properly on `data.cs.purdue.edu`.** After logging into `data.cs.purdue.edu` (physically go to the lab or use ssh remotely, as you are all granted the accounts to CS data machines during this class), please follow these steps to submit your assignment:

1. Make a directory named *'yourname_yoursurname'* (all letters in lower case) and copy all of your files there: `report.pdf`, `decisiontree.py`, `knn.py` and `README` (optional) shall reside directly in this directory. **Don't put the dataset file `winequality-white.csv`.**
2. While in the upper level directory (if the files are in /homes/dan/dan_goldwasser, go to/homes/dan), execute the following command:

   ```
   turnin -c cs578 -p HW1 *your_folder_name*
   ```

   (e.g. your instructor would use: `turnin -c cs578 -p HW1 dan_goldwasser` to submit his homework)

   Keep in mind that old submissions are overwritten with new ones whenever you execute this command.

3. You can verify the contents of your submission by executing the following command:

   ```
   turnin -v -c cs578 -p HW1
   ```

   Do **not** forget the -v flag here, as otherwise your submission would be replaced with an empty one.

Failure to follow the above instructions will incur the penalty when your homework is being graded.

4.3 Late Policies

As declared in the class.

4.4 Plagiarism Policies

Seriuosly, no cheating. If plagiarism was found, both you and the one whose homework you "referred to" receive 0 point on that homework. If you were found to have plagiarised more than once, we will report to the instructor and the department (and your home department if you are not a CS student).