CSC 480/680 – Assignment 1

Individual Assignment

Part 1: Due on September 23, 2020 (70%)

Part 2: Due on October 6, 2020 (30%)

Please read this assignment carefully early on. As you will see, it cannot be done overnight. Please pace yourself accordingly in order to be successful!!!!

The purpose of this assignment is to give you some experience with:

- Different types of classifiers: Decision Trees, Multi-Layer Perceptrons, Convolutional Networks
- Data Manipulation (you will be modifying the data sets slightly)
- The application of filters on your data sets for Feature Selection and the Class Imbalance problem
- Seeing the effect of working on a multi-class classification problem versus binary problems
- Seeing the effect of using a lot of data per class versus using less data per class
- Classifier evaluation

There are two parts to this assignment. In the first part, we will study the issue of overfitting using Decision Trees and Multi-Layer Perceptrons. In the second part, we will examine the power of deep learning on image and text data. The first part should use Scikit-Learn (although, the instructor can allow the use of other packages such as R packages, WEKA in certain cases that need to be cleared first). Because the second part will need more computational resources than the first part, due to the use of a deep learning library, it will be done in the Google Colab environment which gives you free access to GPU computing. For Part 2 of the assignment, you will use the Tensorflow library as the basic machine learning library which is appropriate for larger scale computations than scikit-learn. Keras is a deep learning API that runs on top of Tensorflow, which itself is pre-installed in Google Colab.

Part I

In this part of the assignment, you will be working with two data sets:

- Steel-Plates Fault (https://www.openml.org/d/1504) **multi-labels**
- Ozone-Level-8-hours (https://www.openml.org/d/1487) **binary class**

And two classifiers:

- Decision Trees
- Multi-Layer Perceptrons

You will be using 10-fold Cross-Validation for all your experiments and reporting your results using the AUC. Please try to optimize the classifiers prior to running the experiments.

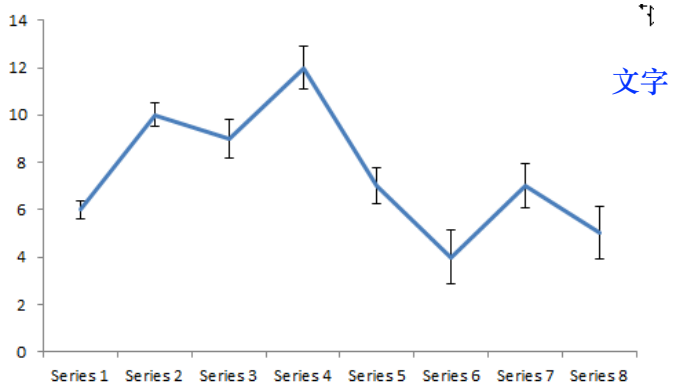
The three research questions I would like you to answer are:

1. At which points (in terms of training set-size and classifier) do we observe overfitting in each data set?
2. Can overfitting be overcome by feature selection? (you can answer this question using the Chi-square feature selection method for example, but it would be even better if you ran some experiment to see which feature selection algorithm works best in your data and use that one. Please use information in: https://scikit-learn.org/stable/modules/feature_selection.html)
3. Are either of your datasets imbalanced? Select the most imbalanced one and show the results obtained using the following four methods on the full data set:
    - Do nothing
    - Randomly oversample the minority class
    - Randomly undersample the minority class
    - Use SMOTE to oversample the minority class.

    You will need to use Python libraries especially for the SMOTE experiment. These links can be helpful: https://www.kaggle.com/qianchao/smote-with-imbalance-data and https://machinelearningmastery.com/smote-oversampling-for-imbalanced-classification/

    What can you say about the different methods used with regard to the overfitting of the minority class? [You may want to use different metrics to make your point; Also, please make sure that you experiment with a few different sizes of instances you add to or remove from the original data set. Select the optimal, but as always in experimental work, do report the values you tried and the results they gave you.

For questions 1 and 2, please report your results in the following format for each data set (ignore the labels on the axes in the figure below: you will use different ones):



文字

graph: get the result
> plot in PPT, excel, word…
> program in Python

On the x-axis, you should indicate the size of the data set. On the y-axis, you should indicate the AUC. Each point should correspond to the result obtained on your 10-fold cross-validation experiment for the particular data-set size indicated on the x-axis. The confidence interval should represent the confidence interval obtained by cross-validation at that point. Questions 1 and 2 should produce two such curves each

ROC curve
ROC Analysis

(i.e., your answer to the first two questions should be 4 graphs of the type shown above). In addition to the graphs, please add a paragraph or two discussing your observations (i.e., what you expected [and the reasons why you expected that] and what you observed, as well as what your observations mean).

For question 3, please report your results in a table of the following format:

	AUC	F1-Measure	G-Mean	Precision	Recall
Do Nothing					
Random Oversampling					
Random Undersampling					
SMOTE					

If you find that reporting metrics other than the ones suggested in the template table above is better, please add them to your table (or substitute the metrics you find least useful for those you find more useful). As for questions 1 and 2, please add a paragraph or two to discuss your observations and what they mean.

Part 2

In this part of the assignment, you will work in two application domains that are very well served by Deep learning: Computer Vision and Natural Language Processing. In particular, we will consider the problems of image and text classification. You will be working with two datasets:

Image classification:

- Cifar10
    - See https://www.openml.org/d/40926

Text classification:

- Sentiment Labelled Sentences Data
    - See https://archive.ics.uci.edu/ml/datasets/Sentiment+Labelled+Sentences

Four useful tutorials can be found at (but feel free to look for others if they help you):

- https://machinelearningmastery.com/when-to-use-mlp-cnn-and-rnn-neural-networks/
- https://colab.research.google.com/github/lexfridman/mit-deep-learning/blob/master/tutorial_deep_learning_basics/deep_learning_basics.ipynb
- https://realpython.com/python-keras-text-classification/
- https://cezannec.github.io/CNN_Text_Classification/#:~:text=1D%20Convolutions,in%20only%20one%20dimension%3A%20time.

The purpose of this part of the assignment is to compare the performance of MLP networks to CNN networks in the context of an image classification problem and a sentiment analysis problem (classify a

sentence as carrying a positive sentiment (e.g., “I love chocolate”) or negative sentiment (e.g., “his performance was substandard on this task”).

A. Image classification
    a. Choose three classes of images to work on.
    b. Transform the images into 1) a vector representation appropriate for MLP; 2) a tensor representation for a 2D-CNN.
    c. Use one-hot-encoding to encode the output for the three classes
    d. Divide the data into a training and a test set (80% for training; 20% for testing. Make sure that the data is chosen at random)
    e. Divide the training data into a training set and a validation set. Optimize MLP’s parameters.
    f. Return your results on the training and testing set. Beware of overfitting and experiment with early stopping to see if that helps decrease the amount of overfitting.
    g. Optimize CNN’s parameters
    h. Return your results on the training and testing set. Same comments about overfitting.
    i. Your report should include:
        i. A table showing the results obtained by MLP and CNN on the training and testing set; You can use accuracy.
        ii. A discussion of your observations (which should discuss the observations made as you experimented and observations about the results)

B. Text Classification
    a. Create a tf-idf vector representation for the text data (Representation 1)
    b. Use a pre-trained Word2Vec embedding to encode your data (Representation 2)
    c. Divide the data, at random, into training and testing sets (Use a 80%-20% divide and use the same division for the two representations)
    d. Run MLP on the training set encoded using tf-idf (optimize the network on the training data (divide the training data into a training and a validation set) as well as you can)
    e. Train a 1-D CNN on the Word2Vec embedding (optimize the network on the training data (divide the training data into a training and a validation set) as well as you can)
    f. Return all the results in a table showing the Precision, Recall and F1-measures. One row should show the tf-idf/MLP combination and one row should show the Word2Vec/1D-CNN combination.
    g. Discuss your observations (which should discuss the observations made as you experimented and observations about the results)

Good luck!!!! And start working on this early!!!!!