Computer Science 3202/6915
Assignment 1 – Understanding Overfitting and the pitfall of testing on the training data

Goal

Understand overfitting and the over-estimation of performance when assessing model performance on training data.

Due date: Sunday February 13th by 11:30pm.

Specifications

Your program should be called A1_overfit.py and it should run in Linux. You are allowed to use functions available in packages such as Scikit-learn, Pandas or NumPy. Your program should take one command-line argument: A filename specifying a tab-delimited plain-text file containing the data. For example, we should be able to execute your program as follows:

```
$python3 A1_overfit.py fileWithData.txt
```

where the $ indicates the terminal prompt and fileWithData.txt is a tab-delimited text file following the format described in the next paragraph.

The data you will use is provided in Brigthspace (filename is A1_data.tsv). This is a tab-delimited text file containing 100 observations. The file has a column header. The first 10 columns are the features and the last column is the output. However, your program should be able to handle data with any number of observations, features or classes following the same format. We will likely test your program on a different data set.

Functionality

Your program should do the following:

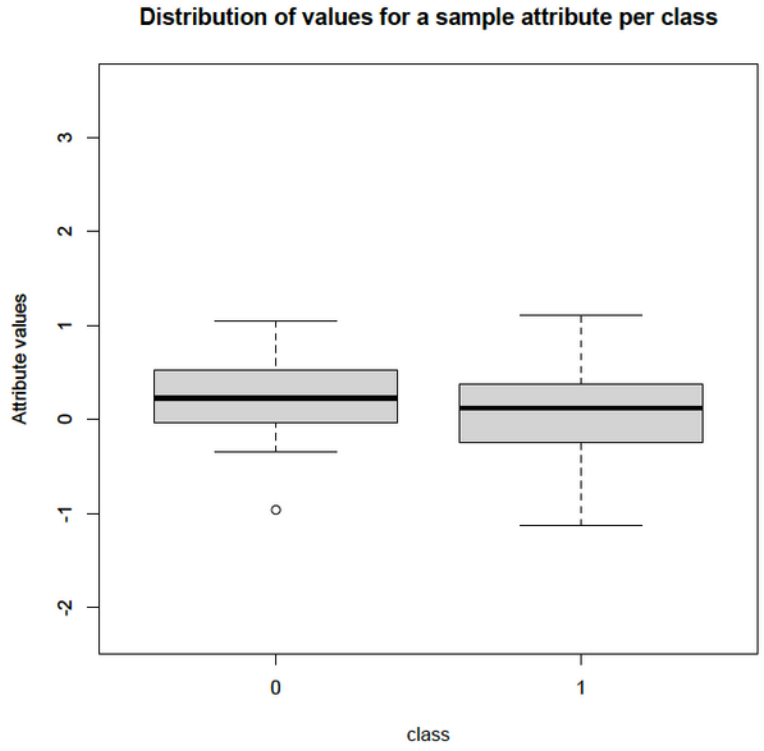
1. Read the command-line argument.
2. Read the input data. You can assume the input file is in the working directory.
3. Randomly select U= 25 instances to be the instances in UnInstance (refer to KNN Algorithm for Classification in Lecture 2).
4. For number of neighbors k = {1, 3, 5, 7, 9},
    1. Perform KNN Algorithm for Classification to predict the class of the instances in UnInstance **using all instances in the data as the training data**. Use the scikit-learn function KNeighborsClassifier. You can choose any appropriate distance metric.
    2. Obtain the accuracy of the classifier. That is the number of instances in UnInstance for which the actual class is equal to the predicted class divided by U.
5. Remove the instances in UnInstance from the input data. The remaining instances are the training data T.
6. For number of neighbors k = {1, 3, 5, 7, 9},
    1. Perform the KNN Algorithm for Classification to predict the class of the instances in UnInstance using only the instances in T as the training data. Use the scikit-learn function KNeighborsClassifier. Use same distance metric chosen in step 4.1.
    2. Obtain the accuracy of the classifier. That is the number of instances in UnInstance for which the actual class is equal to the predicted class divided by U.
7. Print in the screen the value of k followed by a tab (\t) and then the accuracy calculated in steps

4.2 and in 6.2.

Note: To understand better what is going on, you might want to run your program a couple of times with different random seed.

Report

1. The data for this assignment was created by generating 10 normally distributed variables as the attributes, and one randon variable following the binomial distribution with equal probability of 0s or 1s as the output.
   The plot below illustrates the relationship (or lack of) between one of these attributes and the output.



   1. Is there an informative feature? Why? (Hint: To answer this you might want to generate box plots as the one above for each feature).
   2. Do these data contain a "learnable" pattern? Why?
   3. What level of predictive performance in terms of accuracy would you expect from a ML model on these data?

2. Write the performance table

Number of neighbours (k)	Accuracy obtained step 4.2	Accuracy obtained step 6.2

1		
3		
5		
7		
9		

1. Based on the accuracy obtained in step 4.2, what is the best performing model? Briefly explain why this model is performing so well.
2. Why is there a drop in accuracy in step 6.2 compared with step 4.2? If you were to run step 6.2 many times and obtain average accuracy per number of neighbours, what mean accuracy would you expect? Why?
3. Based on the accuracy obtained in step 6.2, is there a best performing model? If yes, which one? Briefly explain why you chose this model as the best performing. Suggest an estimate of the generalization error of this model trained on these data.
4. Which two models have the lowest bias and highest variance? Why?

Submission
Submit through Brightspace the following files (one submission per team). Submit the files individually. That is, do not submit a compressed file.

a) Your python code in a single file called A1_overfit.py
b) A PDF file called A1_report.pdf containing the accuracy table and brief answers to the questions listed under the **Report** section above.
c) Optional – bonus point: a PNG of JPG image with a plot showing the test error cuve and train error curve as a function of the number of neighbours. Remember error is the same as the loss or 1 – accuracy. Plot should contain axis labels and a caption explaining the curves.

Common pitfalls to avoid (i.e., DO NOT do the following or points will be deducted):

1. Submit files in a compressed file.
2. Hardcode filename of the input.
3. Forget to test your program in linux (i.e., your program fails to run in linux).
4. Miss to answer some of the questions in the PDF file.
5. Forget to acknowledge a collaboration or source.
6. Fail to follow specifications.
7. Modify the input data (i.e., your program fails to run with the original data).
8. Hardcode the number of observations in the input data.
9. Hardcode the number of attributes in the input data.

To access LabNet:
From home go to https://remote.labnet.mun.ca/ and select “Linux General Access”.

For on campus access see https://www.labnet.mun.ca/