Assignment 1: Getting Started with Machine Learning

COMP 551 Winter 2024, McGill University
Contact TAs: Huiliang Zhang and Valliappan CA

Please read this entire document before beginning the assignment.

Preamble

- This assignment is **due on January 31st at 11:59pm (EST, Montreal Time).**
- For late submission, 2^k percent will be deducted per k days of the delay.
- To use your 6-day quota as a team, submit your request by emailing `comp551.socs@mcgill.ca` with subject title "A1 extension request" and in the email body specify the number of days (max at 6 days) you need to submit your assignment. You can only submit the request ONCE. Once you request the days, the quota of every member on the team will be reduced by the days you requested even if you end up submitting your assignment prior to the extended deadline. Therefore, plan and use your quota wisely.
- This assignment is to be completed in groups of three. All members of a group will receive the same grade except when a group member is not responding or contributing to the assignment. If this is the case and there are major conflicts, please reach out to the contact TA or instructor for help and flag this in the submitted report. Please note that it is not expected that all team members will contribute equally. However every team member should make integral contributions to the assignment, be aware of the content of the submission and learn the full solution submitted.
- You will submit your assignment on MyCourses as a group. You must register your group on MyCourses and any group member can submit. See MyCourses or here for details.
- We recommend to use **Overleaf** for writing your report and **Google colab** for coding and running the experiments. The latter also gives access to the required computational resources. Both platforms enable remote collaborations.
- You should use Python for this and all assignments. You are free to use libraries with general utilities, such as matplotlib, numpy and scipy for Python, unless stated otherwise in the description of the task. In particular, in most cases you should implement the models

and evaluation functions yourself, which means you should not use pre-existing implementations of the algorithms or functions as found in SciKit learn, and other packages. The description will specify this in a per case basis.

1 Background

In this assignment you will implement two classification techniques — K-Nearest Neighbour (KNN) and Decision Trees (DTs) — and compare these two algorithms on two distinct health datasets. The goal is to get started with programming for Machine Learning, how to properly store the data, run the experiments, and compare different methods. You will also gain experience implementing these algorithms from scratch and get hands-on experience comparing performance of different models.

2 Task 1: Acquire, preprocess, and analyze the data

Your first task is to acquire the data, analyze it, and clean it (if necessary). We will use two fixed datasets in this assignment, outlined below.

- **Dataset 1: NHANES_age_prediction.csv (National Health and Nutrition Health Survey 2013-2014 (NHANES) Age Prediction Subset):**
  `https://archive.ics.uci.edu/dataset/887/national+health+and+nutrition+health+survey+2013-2014+(nhanes)+age+prediction+subset`
- **Dataset 2: Breast Cancer Wisconsin (Original) dataset:**
  `https://archive.ics.uci.edu/dataset/15/breast+cancer+wisconsin+original`

The essential subtasks for this part of the assignment are:

1. Load the datasets into NumPy or Pandas objects in Python.
2. Clean the data. Are there any missing or malformed features? Are there any other data peculiarities that need to be dealt with? **You could remove any examples with missing or malformed features and note this in your report.** This is a straightforward way to handle this issue by simply eliminating missing values. You can use the X.isna() or X.isnull() functions to check for missing data in the dataset, and use X = X.dropna() to eliminate them. You are welcome to explore other possible ways (e.g. imputating missing values with the average of the observed values for the same features)
3. Compute basic statistics on the data to understand it better. For example, what are the mean of each feature for the positive group and the mean of each feature for the negative group? If you rank the squared difference of the group means, are the top features known to be associated with the target variable?

3 Task 2: Implementing KNN and DT

You are free to implement these models as you see fit, but you should follow the equations that are presented in the lecture slides, and you must implement the models from scratch (i.e., you **CANNOT** use SciKit Learn or any other pre-existing implementations of these methods). However, you are free to use relevant code given at the course GitHub `https://github.com/yueliyl/comp551-notebooks`.

Specifically, your two main sub-tasks in this part are to:

1. Implement KNN with appropriate distance function.
2. Implement DT with appropriate cost function.

You are free to implement these models in any way you want, but you must use Python and you must implement the models from scratch (i.e., you cannot use SciKit Learn or similar libraries). Using the NumPy or Pandas package, however, is allowed and encouraged. Regarding the implementation, we recommend the following approach (but again, you are free to do what you want):

- Implement both models as Python classes. You should follow the Object Oriented Programming (OOP) paradigm. Use the Constructor for the class to initialize the model parameters as attributes, as well as to define other important properties of the model.
- Each of your models classes should have (at least) two functions:
  - Define a `fit` function, which takes the training data (i.e., $\mathbf{X}$ and $\mathbf{y}$) — as well as other hyperparameters (e.g., the value of K in KNN and maximum tree depth in DT) — as input. This function should train your model by modifying the model parameters.
  - Define a `predict` function, which takes a set of input points (i.e., $\mathbf{X}$) as input and outputs predictions (i.e., $\hat{\mathbf{y}}$) for these points.
- In addition to the model classes, you should also define a function `evaluate_acc` to evaluate the model accuracy. This function should take the true labels (i.e., $\mathbf{y}$), and predicted labels (i.e., $\hat{\mathbf{y}}$) as input, and it should output the accuracy score.

4 Task 3: Running experiments

The goal of this assignment is to have you compare different features and models.

Split each dataset into training and test sets. Use test set to estimate performance in all of the experiments after training the model with training set. Evaluate the performance using accuracy and Area Under the Receiver Characteristic Curve (AUROC). For computing the AUROC, you are allowed to use `sklearn.metrics.roc_curve` function.

You are welcome to perform any experiments and analyses you see fit (e.g., to compare different features), **but at a minimum you must complete the following experiments in the order stated below and describe your findings for each of them**:

1. Compare the accuracy and AUROC of KNN and DT algorithm on the two datasets.
2. Test different K values and see how it affects the training data accuracy and test data accuracy of KNN.
3. Similarly, check how maximum tree depth can affect the performance of DT on the provided datasets.
4. Try out different distance/cost functions for both models.
5. Plot the ROC for KNN and DT on the test data. Both ROC curves need to be plotted on the same plot to enable comparison between of the method performance. Also show the AUROC of each method in the figure legend. You may use evaluation code from the ModelEvaluationAndSelection.ipynb colab (but not the implementation of scikit-learn KNN and DT).
6. Describe how you obtain the key features used in KNN (e.g., external feature selection by correlation with the labels).
7. For DT, you can compute a *rough* feature importance score for each feature d by counting the number of non-leaf nodes where feature d is used. Report what the top 5 most important features. Are they the same as the simple mean difference approach described in the subtask 3 in Section 2? If not, why?

Note: The above experiments are the minimum requirements that you must complete; however, this assignment is open-ended. Here are a few suggestions:

- You can split the data into training, validation and testing and use the validation set to select the best K and the best tree depth and evaluate the best choice on the test set.
- You may perform K-fold cross-validation.
- You may also improve the model performance by implementing weighted KNN as we discussed in class.
- For DT, improve the feature importance score for each feature d by a weighted sum based on the reduction of cost (e.g., gini-index at node j):

$$R_d = \sum_{j=1}^{J} \Delta g_j \mathbb{I}[v_j = d]$$

where $\Delta g_j = (g_j - g_j(left)) + (g_j - g_j(right))$

You do not need to do all of these things, but you should demonstrate creativity, rigour, and an understanding of the course material in how you run your chosen experiments and how you report on them in your write-up.

5 Deliverables

You must submit two separate files to MyCourses (**using the exact filenames and file types outlined below**):

1. `assignment1_group-k.ipynb`: Your data processing, classification and evaluation code should be all in one single Jupyter Notebook. Your notebook should reproduce all the results in your reports. **Please ensure that all the original training and output results are saved in your notebook, and these should be the same results provided in your write-up.** The TAs may run your notebook to confirm your reported findings.
2. `assignment1_group-k.pdf`: Your (max 5-page) assignment write-up as a pdf (details below).

where k is your group number.

5.1 Assignment write-up

Your team must submit a assignment write-up that is a maximum of five pages (single-spaced, 11pt font or larger; minimum 0.5 inch margins, an extra page for references/bibliographical content can be used). We highly recommend that students use LaTeX to complete their write-ups. **This first assignment has relatively strict requirements, but as the course progresses your assignment write-ups will become more and more open-ended**. You have some flexibility in how you report your results, but you must adhere to the following structure and minimum requirements:

Abstract (100-250 words) Summarize the assignment task and your most important findings. For example, include sentences like "In this assignment we investigated the performance of two machine learning models on two benchmark datasets", "We found that the Decision Tree approach achieved worse/better accuracy than K - Nearest Neighbour."

Introduction (5+ sentences) Summarize the assignment task, the two datasets, and your most important findings. This should be similar to the abstract but more detailed. You should include background information and citations to relevant work (e.g., other papers analyzing these datasets).

Methods (4+ sentences) Briefly describe the general algorithmic concepts (not the code) of the machine learning methods you implemented (i.e., KNN and DT) *in your own words*. Your description can be paraphrased from but not identical to those in the textbooks.

Datasets (5+ sentences) Very briefly describe the datasets and how you processed them. Present the exploratory analysis you have done to understand the data, e.g. class distribution.

Results (7+ sentences, possibly with figures or tables) Describe the results of all the experiments mentioned in **Task 3** (at a minimum) as well as any other interesting results you find (Note: demonstrating figures or tables would be an ideal way to report these results).

Discussion and Conclusion (5+ sentences) Summarize the key takeaways from the assignment and possible directions for future investigation.

Statement of Contributions (1-3 sentences) State the breakdown of the workload across the team members.

6 Evaluation

The assignment is out of 100 points, and the evaluation breakdown is as follows:

- Completeness (20 points)
  - Did you submit all the materials?
  - Did you run all the required experiments?
  - Did you follow the guidelines for the assignment write-up?
- Correctness (40 points)
  - Are your models implemented correctly?
  - Are your reported accuracies close to our solution?
  - Do you observe the correct trends in the experiments (e.g., how the accuracy and AUROC changes as the K values of KNN or maximum depth of DT increases)?
  - Do you observe the correct impact of different distance/cost functions on model performance?
  - Do you find meaningful features with high feature importance scores based on the trained DT?
- Writing quality (30 points)
  - Is your report clear and free of grammatical errors and typos?
  - Did you go beyond the bare minimum requirements for the write-up (e.g., by including a discussion of related work in the introduction)?
  - Do you effectively present numerical results (e.g., via tables or figures)?
- Originality / creativity (10 points)
  - Did you go beyond the bare minimum requirements for the experiments?

- **Note:** Simply adding in a random new experiment will not guarantee a high grade on this section! You should be **thoughtful and organized** in your report. That is, the distinctive ideas that you came up with should blend in your whole story. For instance, explaining the motivations behind them would be a great starting point.

7 Final remarks

You are expected to display initiative, creativity, scientific rigour, critical thinking, and good communication skills. You don't need to restrict yourself to the requirements listed above - feel free to go beyond, and explore further.

You can discuss methods and technical issues with members of other teams, but **you cannot share any code or data with other teams.**