

MiniProject 1: Getting Started with Machine Learning

COMP 551, Winter 2022, McGill University
Contact TAs: Xiru Zhu and Venkata Naga Sai Praneeth

Please read this entire document before beginning the assignment.

Preamble

- This mini-project is **due on February 8th at 11:59pm (EST, Montreal Time)**. There is a penalty of 2^k percent penalty for k days of delay, which means your grade will be scaled to be out of $100 - 2^k$. No submission will be accepted after 6 days of delay.
- This mini-project 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 project. If this is the case and there are major conflicts, please reach out to the group TA 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 project, 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 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 the following mini-projects. 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.

Background

In this miniproject you will implement two classification techniques—**K-Nearest Neighbour and Decision Trees**—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.

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 project, outlined below.

- **Dataset 1: hepatitis.csv (Hepatitis dataset):**

<http://archive.ics.uci.edu/ml/datasets/Hepatitis>

- **Dataset 2: messidor_features.arff (Diabetic Retinopathy Debrecen dataset):**

<https://archive.ics.uci.edu/ml/datasets/Diabetic+Retinopathy+Debrecen+Data+Set>

The essential subtasks for this part of the project 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 are other data oddities that need to be dealt with? **You should remove any examples with missing or malformed features and note this in your report.**

If you choose to play with Pandas dataframes, a handy line of code that might be helpful is `df[df.eq('').any(1)]`, where `df` is the dataframe, and `''` represents a missing value in the datasets. This is a straightforward way to handle this issue by simply eliminating rows with missing values. You are welcome to explore other possible ways!

3. Compute basic statistics on the data to understand it better. E.g., what are the distributions of the positive vs. negative classes, what are the distributions of some of the numerical features?

Task 2: Implement the models

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 in the course website.**

In particular, your two main tasks in the part are to:

1. Implement K - Nearest Neighbour .
2. Implement Decision Tree 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 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., **X** and **Y**)—as well as other hyperparameters (e.g., K value in KNN and maximum tree depth in Decision Tree)—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., **X**) as input and outputs predictions (i.e., \hat{y}) for these points.
- In addition to the model classes, you should also define a functions `evaluate_acc` to evaluate the model accuracy. This function should take the true labels (i.e., **y**), and target labels (i.e., \hat{y}) as input, and it should output the accuracy score.

Task 3: Run experiments

The goal of this project 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. 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:**

1. Compare the accuracy of KNN and Decision Tree algorithm on the two datasets.
2. Test different K values and see how it affects the training data accuracy and test data accuracy.
3. Similarly, check how maximum tree depth can affect the performance of of Decision Tree on the provided datasets. Describe your findings.
4. Try out different distance/cost functions for both models. Describe your findings.
5. Present a plot of the decision boundary for each model. Describe the key features in short.

Note: The above experiments are the minimum requirements that you must complete; however, this project is open-ended. For example, you might investigate different stopping criteria for Decision Tree or different features that you select for the training process. We would also love to see possible ways to improve model performance. 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.

Deliverables

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

1. **code.zip**: Your data processing, classification and evaluation code (as some combination of .py and .ipynb files).
2. **writeup.pdf**: Your (**max 5-page**) project write-up as a pdf (details below).

Project write-up

Your team must submit a project 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 mini-project report has relatively strict requirements, but as the course progresses your project 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 project task and your most important findings.** For example, include sentences like “In this project 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 project 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).

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. Highlight any possible ethical concerns that might arise when working these kinds of datasets.

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 project and possibly directions for future investigation.

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

Evaluation

The mini-project 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 project 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 changes as the maximum depth of Decision Tree increases)?
 - Do you observe the correct impact of different distance/cost functions on model performance?
 - Do you find notable features of the decision boundaries?
- Writing quality (25 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 (15 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 triggers behind them would be a great starting point.

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.**

Congratulations on completing your first course project! You are now capable of making simple diagnosis of disease based on patients' health data. This is one of the most significant applications of classification algorithms. As the class goes on, we will see more machine learning models and their interesting applications in real life.