CSC242 Intro to AI
Project 4: Learning

In this project you will have the opportunity to implement and evaluate one or more machine learning methods. This is a huge area of application and research, and is well-covered in upper-level Computer Science courses. It's also nearing the end of the term. So we can really only scratch the surface with this project. But remember, the more you do and the more you do yourself, the more you'll learn.

For the project, you MUST implement linear classifiers as described in AIMA Sec. 18.6 (3rd edition) and in class. Linear classifiers are the basis of neural networks (AIMA 18.7, also 3rd edition), so for extra credit (max 20%) you may also implement neural networks. Both aspects of the project are described in detail below.

For this project, you MUST also produce a short report describing what you did and presenting the results of your learning program(s). The requirements for this are also detailed below. It is more than a README.

Linear Classifiers

Linear classifiers are well covered in AIMA Sect. 18.6 (3rd edition).

Think about what it takes to represent a linear classifier and the data used to train it in a computer program. **THINK ABOUT IT NOW**.

Ok. I hope you thought about things like input vectors, outputs, weight vectors, and update rules. None of these are hard to implement, but you should think through the design before jumping in with unstructured code.

You will learn the most if you develop your implementation yourself. If you need a little help getting started, look at the documentation for the code I have provided from package "`lc`". It will suggest some classes and give you their APIs. That may be enough for you to write the code. But if you need a bit more help, you can build off the code I have provided. You will need to implement the crucial classes and/or methods for actually learning the linear classifiers.

You MUST implement both a perceptron classifier (18.6.3) and a logistic classifier (18.6.4). Almost all of the code can be shared if you design it right. You should be able to replicate

something like the textbook results for the earthquake problem (Figures 18.15, 18.16, and 18.18).

Demonstrate your program on the earthquake data (both clean and noisy datasets) and the "house votes" dataset ("numerical" version), all of which are provided in our code bundle. Feel free to try other datasets also. See below for details regarding your report.

Neural Networks

Neural networks are covered in AIMA Section 18.7 (3rd edition). It does cover all the important definitions for both single-layer and multi-layer feed-forward networks, and it provides the algorithm for backpropagation in multi-layer networks (Fig. 18.24). That said, the presentation is very concise. So if you choose to implement this type of learner, be prepared to do some thinking and/or additional research as you develop and evaluate your system.

It isn't hard to think about what you need to represent a neural network in a computer program. **THINK ABOUT IT NOW**.

Ok. I hope you thought about "units," layers, connections, weights, activation functions, inputs, and outputs. Remember that a neural network is simply a graph of linear classifiers (typically using a logistic threshold). It is not hard to design classes incorporating these elements. However I suggest that you understand how the backpropagation algorithm works before you lock in your design. In particular, note that it requires that you be able to go both forward and backward through the layers of your networks, even though the network is "feed-forward."

As always, you will learn the most if you develop your implementation yourself. If you need a little help, look at the documentation for the code I have provided from the package "`nn`". That will give you some suggestions for classes and APIs. And if you need more help, you can build off the code I have provided. You will need to implement the crucial classes and/or methods for actually learning the linear classifiers in `nn.core`.

If you choose to do this (for extra credit), you MUST implement a multi-layer network with hidden units. A single-layer network is essentially a set of one or more linear classifiers. A multi-layer network is a "true" neural network that must be trained using backpropagation.

Demonstrate your program on the Iris dataset and the MNIST dataset (handwritten digit recognition). Files, or pointers to data files in the case of MNIST, are provided in our

code bundle. For the Iris dataset, try a two-layer network with four inputs, seven hidden units, and three output units (one for each species of Iris). For MNIST, visit `http://yann.lecun.com/exdb/mnist/` and try some of the networks described there. For example, a two-layer network with 300 or 1000 hidden units, or a three-layer network with 500 and 150 units (first and second layer, respectively). As the textbook says: "Unfortunately, for any *particular* network structure, it is harder to characterize exactly which functions can be represented and which ones cannot."

Feel free to try other datasets also. See below for details regarding your report.

Datasets for Machine Learning

There are many, many datasets available online for training different types of machine learning systems. One of the best sources is the UCI Machine Learning Archive: `http://archive.ics.uci.edu/ml`. Some of the datasets from this archive are downloadable with the project description.

There is always some work reading these files and getting them into the proper form for your learning system. It's easy for simple datasets, like the "Iris" dataset, and harder for more complicated datasets. For problems based on images (or other media), there is often a dataset with a set of attribute values extracted from the data already available, so that you can focus on machine learning rather than image processing. Or you may be interested in extracting the attributes from the media yourself. Note that continuous-valued datasets may need to be discretized for some types of learning (unless you want to implement more sophisticated algorithms).

Start simple. Use the earthquake dataset for linear classifiers, or the Iris dataset for neural networks, to get started. They are manageable. Then try something bigger and more interesting.

Report Requirements

For this project, your report is not simply a README.

The first section of your report MUST include the exact commands needed to build your programs, if they need to be built.

Linear Classifiers

For the required linear classifiers, your programs' output must include data sufficient to produce *training curves* as seen in AIMA Figs. 18.16 and 18.18 (3rd edition). And then you must use that output to produce the graphs and include them in your report (clearly labelled and explained). You may use Excel or Google Docs or `gnuplot` or make your own plotter for this.

Your report MUST include the commands necessary to produce the data for all six graphs from AIMA for the earthquake data, and include the graphs themselves. For the "house votes" dataset, do something similar and include the commands and the graphs in your report.

Include a short paragraph (not bullet points) describing the graphs and discussing the results. It doesn't have to be much, but it has to be readable. Bullet points or no text at all will get a lower grade.

Note: The graphs for the perceptron classifier, shown in AIMA Fig. 18.16, are labelled "Proportion Correct," meaning accuracy, which should go towards 100% as the classifier is trained.

However the graphs for the logistic classifier, shown in AIMA Fig. 18.18, are labelled "squared error per sample." Assuming that this is the Mean Squared Error (MSE), it is the mean L_2 loss over all the samples (examples). But the error won't go up as the classifier is trained, so I assume that they are actually plotting $1.0 - \mathrm{MSE}$, which is 1.0 if the classifier classifies the examples perfectly.

Label your graphs properly!

Neural Networks

If you choose to implement neural networks, your program's output should include, for each problem, the overall accuracy (on the training data) after training for some number of epochs, for example $N = 1000$.

You should also perform k-fold cross-validation for some reasonable k such as $k = 10$ and output the results of each trial and the overall average accuracy.

Finally, your program(s) should try training for a varying number of epochs from 100 to 3000 by 100s (or something else reasonable) and output data sufficent to produce

curves similar to those shown in AIMA Fig. 18.25 (3rd edition). Measure accuracy on the training data (Fig. 18.25(a)) and error on the test set (Fig. 18.25(b)). The former should go up, the latter should go down, just like the AIMA figures.

Include a short paragraph (not bullet points) discussing these results along with the graphs. Again, bullet points or no text at all will get a lower grade.

Additional Requirements and Policies

The short version:

- You may use Java, C/C++, or Python. I STRONGLY recommend Java.
- You MUST use good object-oriented design (yes, even in Python).
- There are other language-specific requirements detailed below.
- You must submit a ZIP including your source code, a README, and a completed submission form by the deadline.
- You must tell us how to build your project in your README.
- You must tell us how to run your project in your README.
- Projects that do not compile will receive a grade of **0**.
- Projects that do not run or that crash will receive a grade of **0** for whatever parts did not work.
- Late projects will receive a grade of **0** (see below regarding extenuating circumstances).
- You will learn the most if you do the project yourself, but collaboration is permitted in groups of up to 3 students.
- Do not copy code from other students or from the Internet.

Detailed information follows. . .

Programming Requirements

- You may use Java, C/C++, or Python for this project.
  - I STRONGLY recommend that you use Java.
  - Any sample code we distribute will be in Java.
  - Other languages (Haskell, Clojure, Lisp, *etc.*) only by prior arrangement with the instructor.
- You MUST use good object-oriented design.
  - In Java, C++, or Python, you MUST have well-designed classes.
  - Yes, even in Python.
  - In C, you must have well-designed "object-oriented" data structures (refresher: C for Java Programmers guide and tutorial)
- No giant `main` methods or other unstructured chunks of code.
- Your code should use meaningful variable and function/method names and have plenty of meaningful comments. But you know that...

Submission Requirements

You MUST submit your project as a ZIP archive containing the following items:

1. The source code for your project.
2. A file named `README.txt` or `README.pdf` (see below).
3. A completed copy of the submission form posted with the project description (details below).

Your README **MUST** include the following information:

1. The course: "CSC242"
2. The assignment or project (*e.g.*, "Project 1")
3. Your name and email address

4. The names and email addresses of any collaborators (per the course policy on collaboration)
5. Instructions for building and running your project (see below).

The purpose of the submission form is so that we know which parts of the project you attempted and where we can find the code for some of the key required features.

- Projects without a submission form or whose submission form does not accurately describe the project will receive a grade of **0**.
- If you cannot complete and save a PDF form, submit a text file containing the questions and your (brief) answers.

Project Evaluation

You **MUST** tell us in your README file how to build your project (if necessary) and how to run it.

Note that we will NOT load projects into Eclipse or any other IDE. We **MUST** be able to build and run your programs from the command-line. If you have questions about that, go to a study session.

We **MUST** be able to cut-and-paste from your documentation in order to build and run your code. **The easier you make this for us, the better your grade will be.** It is **your** job to make the building of your project easy and the running of its program(s) easy and informative.

For **Java** projects:

- The current version of Java as of this writing is: 16.0.2 (OpenJDK)
- If you provide a `Makefile`, just tell us in your README which target to make to build your project and which target to make to run it.
- Otherwise, a typical instruction for building a project might be:

  ```
  javac *.java
  ```

  Or for an Eclipse project with packages in a `src` folder and classes in a `bin` folder, the following command can be used from the `src` folder:

```
javac -d ../bin `find . -name '*.java'`
```

- And for running, where `MainClass` is the name of the main class for your program:

  ```
  java MainClass [arguments if needed per README]
  ```

  or

  ```
  java -d ../bin pkg.subpkg.MainClass [arguments if needed per README]
  ```

- You **MUST** provide these instructions in your README.

For **C/C++** projects:

- You MUST use at least the following compiler arguments:

  ```
  -Wall -Werror
  ```

- If you provide a `Makefile`, just tell us in your README which target to make to build your project and which target to make to run it.
- Otherwise, a typical instruction for building a project might be:

  ```
  gcc -Wall -Werror -o EXECUTABLE *.c
  ```

  where `EXECUTABLE` is the name of the executable program that we will run to execute your project.

- And for running:

  ```
  ./EXECUTABLE [arguments if needed per README]
  ```

- You **MUST** provide these instructions in your README.
- Your code should have a clean report from `valgrind`. If we have problems running your program and it doesn't have a clean report from `valgrind`, we are going to assume that your code is wrong. If you are a C/C++ programmer and don't know about `valgrind`, look it up. It is an essential tool.

For **Python** projects:

I strongly recommend that you NOT use Python for projects in CSC242. All CSC242 students have at least two terms of programming in Java. The projects in CSC242 involve the representations of many different types of objects with complicated relationships between them and algorithms for computing with them. That's what Java was designed for.

But if you insist...

- The latest version of Python as of this writing is: 3.9.6 (python.org).
- You must use Python 3 and we will use a recent version of Python to run your project.
- You may **NOT** use any non-standard libraries. This includes things like NumPy. Write your own code—you'll learn more that way.
- We will follow the instructions in your README to run your program(s).
- You **MUST** provide these instructions in your README.

For **ALL** projects:

We will **NOT** under any circumstances edit your source files. That is your job.

Projects that do not compile will receive a grade of **0**. There is no way to know if your program is correct solely by looking at its source code (although we can sometimes tell that is incorrect).

Projects that do not run or that crash will receive a grade of **0** for whatever parts did not work. You earn credit for your project by meeting the project requirements. Projects that don't run don't meet the requirements.

Any questions about these requirements: go to study session **BEFORE** the project is due.

Late Policy

Late projects will receive a grade of **0**. You MUST submit what you have by the deadline. If there are extenuating circumstances, submit what you have before the deadline and

then explain yourself via email.

If you have a medical excuse (see the course syllabus), submit what you have and explain yourself as soon as you are able.

Collaboration Policy

I assume that you are in this course to learn. You will learn the most if you do the projects YOURSELF.

That said, collaboration on projects is permitted, subject to the following requirements:

- Groups of no more than 3 students, all currently taking CSC242.
- You MUST be able to explain anything you or your group submit, IN PERSON AT ANY TIME, at the instructor's or TA's discretion.
- One member of the group should submit code on the group's behalf in addition to their writeup. Other group members MUST submit a README (only) indicating who their collaborators are.
- All members of a collaborative group will get the same grade on the project.

Academic Honesty

I assume that you are in this course to learn. You will learn nothing if you don't do the projects yourself.

Do not copy code from other students or from the Internet.

Avoid Github and StackOverflow completely for the duration of this course.

There is code out there for all these projects. You know it. We know it.

Posting homework and project solutions to public repositories on sites like GitHub is a violation of the University's Academic Honesty Policy, Section V.B.2 "Giving Unauthorized Aid." Honestly, no prospective employer wants to see your coursework. Make a great project outside of class and share that instead to show off your chops.