Naive Bayes Week 2 AMA

Friday, April 9 2021

Tomoko Sakurayama, Jason Xia, Matt Hokinson, Patrick Gallagher, Steven Pan
Objectives

- Implement the classification portion of a machine learning task
  - What good is a model if you can’t use it for its intended purpose?
- Connect your classifier to a Cinder sketchpad
  - Integrating your code with an existing codebase is a common task in software engineering
  - Additional experience with Cinder, which you’ll use on the final project that starts this week!
Common Week 1 Issues - Inefficiencies

Addressing inefficient NB implementations:

- **Suboptimal implementation**
  - Logical structure issues -- pseudocode in next slide
  - Each image should ideally only be iterated through once

- **Returning/iterating over large of objects by copy**
  - Images should be read-only after they are read in
  - Enforce with const ref iteration

  ```cpp
  for (Obj o : objects) { … }
  vs.
  for (const Obj& o : objects) { … }
  ```

- **Restructure your implementation if training time > 60 seconds**
  - You may need to refactor the given pseudocode slightly to integrate it with existing code
  - Your implementation does not have to look exactly like this
**Pseudocode for Training**

```plaintext
TRAINMODEL(IMAGES):
    // All values in vectors initially set to zero.
    priors ← vector[numClasses]
    shadedPixelCounts ← vector[imageDimension][imageDimension][numClasses][numShades]
    for image in images:
        priors[image.label] += 1
        for row ← 0 to imageDimension:
            for col ← 0 to imageDimension:
                shadedPixelCounts[row][col][image.label][image.getShade(row, col)] += 1

    // Transform the counters in the 4D vectors into smoothed conditional probabilities
    // using the formula described in the documentation.
    // Then, do the same for the priors.
```
Common Week 1 Issues Cont.: Abstraction/Encapsulation

Getter example:

```cpp
vector<vector<int>> GetImagePixels();
```

Issues:

- Inflexible
  - Difficult to refactor: Change ints to size_t or bools? Change vector to a map?
- Violating encapsulation
  - Exposes implementation details

Better example:

```cpp
int GetShade(size_t row, size_t col);
```

→ Same idea applies for accessing information of your probability model!
Common Week 1 Issues Cont.:

Instream/Outstream Operator Overloading « « >>

Recap

- **Instream operator »»**
  - Extracts characters from the given stream into a given object
  - Interpret the characters as the representation of a value of the proper type
  - Example: `cin »» mystring;`
  - Example: `cin »» myclass;`

- **Outstream operator « «**
  - Formats and outputs the values of the given object into a given stream
  - Example: `cout « « myint;`
  - Example: `myfile « « myclass;`
Inside class Model:

```cpp
/**
 * This method overloads the >> operator to read in the training data
 */
friend std::istream& operator>>(std::istream& modes_fs, Model&);

/** Builds a model from a file. */
int BuildModel(string filename);

/** Prints a trained model. */
int Print(string filename);

/** Saves a trained model to a file. */
int Save(string filename);

/** Loads a trained model from a file */
int Load(string filename);
```

Issues:

- `>>` is not following expected functionality
- FileIO is happening within the class
- “Print” and “Save” are redundant to:
  - `cout << Model`
  - `file << Model`
- “Load” should really be the instream operator overload
Classification -- Questions to Consider

- What is the relationship between classification and training?
- How can you avoid duplicate code in reading in training data vs classification data?
- As a user, what kind of class interface would you expect out of a classification program?
  - Create well-defined methods with intuitive inputs and return values
Linker Errors

- Largely naming issues
- Error messages may seem cryptic, but if you know where to look, they are pretty descriptive
- There are two main types which are very common, and quite easy to fix
Undefined Reference

- Something like this:
  ```
  /tmp/cc2Q0kRa.o: In function `main':
  /tmp/cc2Q0kRa.o(.text+0x10): undefined reference to `Print(int)'
  collect2: ld returned 1 exit status
  ```

- What this means
  - The linker cannot find the reference, or actual source code, for the function “Print(int)”
    - This might be confusing because you included the header file, so you don’t get an error on compilation
  - Causes:
    - Spelling or a type issue in the parameters
    - Forgetting to write the source code for this function in a .cpp file
  - Note* On many machines, this error might come out as “undefined symbols for architecture x86_64”
Undefined reference to main()

- Special case of the above error
- Means that the linker was unable to find a “main()” function of some sort in your code

Things to check:
- Make sure that you main method is in a file included in CMakeLists.txt and is being compiled into the selected executable
- If working on tests, check for “#define CATCH_CONFIG_MAIN”
Sketchpad Demo
Visit the pinned thread on Campuswire to ask your questions. If you see a question you'd like to hear the answer to, upvote it.