Lab 4

Hide Assignment Information

Instructions

Due: May 2, 10pm
Final Late submissions: May 12, 10pm -- no extensions will be given

Assignment

Write a program demonstrating two supervised learning algorithms: kNN and Naive Bayes'

The two algorithms do not need to share anything in common, but you may wish to do so for the evaluation portion.

Grading

out of 100:

- 45 points each algorithm for correct behavior
  - Your precision and recall numbers should be very close to mine (small discrepancies allowed)
- 10 points for well written code: clear data structures, methods, control flow, etc.

Running your program

Example syntax: (which would be a Naive Bayes with no Laplacian)

```
> learn -train some-file.txt -test other-file.txt
```

Flags:

- -train <file> : the training file (more below)
- -test <file> : the testing data file
- -K : if > 0 indicates to use kNN and also the value of K (if 0, do Naive Bayes')
- -C : if > 0 indicates the Laplacian correction to use (0 means don't use one)
- [optional] a -v verbose flag that outputs each predicted vs actual label (could be useful for testing, but not required)

Note: it is illegal for both K and C to be greater than 0 (they both default to 0)

Inputs (applies to both algorithms)

Both the test and training file will be a csv file where each line represents one 'record'. There will be no header line.

The last entry in a row is the label, and can be any alphanumeric identifier including '_', the rest of the entries are integer extracted features. e.g.

1,5,7,3,ocean_view

In this case there are 4 predictive attributes.

Note: every line in both files should have the same number of columns or it is an error.

Also: There will be no null entries for predictive attributes to worry about.

kNN (using euclidean distance-squared, weighted)

For k-nearest neighbor, there is no training, you just parse and load the file into memory, then use it for classification.

As discussed in class, for each each test point you compute the distance to each training point, picking the K nearest ones and they then

"vote" on the classification using a weight of 1 over the distance. For distance we will use euclidean squared between two points:

For example of there are 3 predictive attributes: $D(<x_1, y_1, z_1>, <x_2, y_2, z_2>) = (z_2-z_1)^2 + (y_2-y_1)^2 + (x_2-x_1)^2$

You then compare the predicted label to the actual one and record for later metrics.

Naive Bayes'

For training you should load the training file into memory, but then compute all of the conditional and pure probabilities in advance (including laplacian smoothing if applicable).

Then when doing predictions on the test file, you do the *argmax* as per class to predict a label, recording versus the actual one for later metrics.

Output

Both algorithms should record: true positives, false positives, and negatives so that you can compute precision and recall.

Once the test set has been evaluated you then print out precision and recall as frequencies in fraction form. e.g. (Also see examples)

Label=A Precision=2/3 Recall=2/3

Label=B Precision=2/3 Recall=2/3

Examples

1. kNN 1 (small) train | test | knn1.out (-K 3)
2. kNN 2 (medium) train | test | knn2.out (-K=3)
3. kNN 3 (large) train | test | knn3.e2.out (-K=3) | knn3.5.out (-K=5) | knn3.7.out (-K=7)

4. Bayes 1 (tiny) train | test | out -C 0 | out -C 0 -v | out -C 1 | out -C 1 -v
5. Bayes 2 train | test | -C 0 | -C 0 -v | -C 1 | -C 1 -v | -C 2 | -C 2 -v

Submission

On Brightspace, a zip file containing:

- Code for the two programs
- Makefile/BUILD/pom.xml/etc needed to build your code
- README indicating how to compile and run your code
- In your README include any classmates you consulted with
- As previously indicated, the code should build and run on department linux machines such as snappy1

Due Date

May 2, 2022 10:00 PM

Submit Assignment

Files to submit *

(0) file(s) to submit

After uploading, you must click Submit to complete the submission.

Add a File **Record Audio** **Record Video**

Comments

Submit **Cancel**