CS 478 - Machine Learning

Projects

Data Representation

Basic testing and evaluation schemes

Programming Issues

- Program in any platform you want
- Realize that you will be doing actual training which can be more time-consuming if implemented in an inefficient way or with an inefficient platform
- We will supply a basic ML toolkit in either C++, Java, or Python – Could MUST be your own! - Learning
- You are welcome to create your own toolkit if you would like, but you need to have at least the level of functionality as in the supplied versions
- Toolkit details found in content section of Learning Suite

Machine Learning Toolkit

- The CS 478 tool kit is intended as a starting place for working with machine learning algorithms. It provides the following functionality to run your algorithms:
  - Parses and stores the ARFF file (ARFF is the data set format we will use)
  - Randomizes the instances in the ARFF file
  - Allows normalization of attributes
  - Parses command line arguments
  - Provides four evaluation methods:
    - Training set method: The model is evaluated on the same data set that was used for training
    - Static split test set method: Two distinct data sets are made available to the learning algorithm; one for training and one for testing
    - Random split test set method: A single data set is made available to the learning algorithm and the data set is split such that *x*% of the instances are randomly selected for training and the remainder are used for testing, where you supply the value of *x*.
    - *N*-fold cross-validation
  - Allows selection of which ML model to train and test with
  - Ability to report training and accuracy information (training and test set accuracies, learning time, etc.)

Gathering a Data Set

- Data Types
  - Nominal (aka Categorical, Discrete)
  - Continuous (aka Real, Numeric)
  - Linear (aka Ordinal) – Is usually just treated as continuous, so that ordering info is maintained
- Consider a Task: Classifying the quality of pizza
  - What features might we use?
- How to represent those features?
  - Will usually depend on the learning model we are using
- Classification assumes the output class is nominal. If output is continuous, then we are doing *regression*.

Fitting Data to the Model

- Continuous -> Nominal
  - Discretize into bins – more on this later
- Nominal -> Continuous (Perceptron expects continuous)
  a) One input node for each nominal value where one of the nodes is set to 1 and the other nodes are set to 0
     - Can also *explode* the variable into n-1 input nodes where the most common value is not explicitly represented (i.e. the all 0 case)
  b) Use 1 node but with a different continuous value representing each nominal value
  c) Distributed – $\log_b n$ nodes can uniquely represent n nominal values (e.g. 3 binary nodes could represent 8 values)
  d) If there is a very large number of nominal values, could cluster (discretize) them into a more manageable number of values and then use one of the techniques above
- Linear data is already in continuous form

Data Normalization

- What would happen if you used two input features in an astronomical task as follows:
  - Weight of the planet in grams
  - Diameter of the planet in light-years

Data Normalization

- What would happen if you used two input features in an astronomical task as follows:
  - Weight of the planet in grams
  - Diameter of the planet in light-years
- Normalize the Data between 0 and 1 (or similar bounds)
  - For a specific instance, could get the normalized feature as follows:

$$f_{normalized} = (f_{original} - Minvalue_{TS})/(Maxvalue_{TS} - Minvalue_{TS})$$

- Use these same Max and Min values to normalize data in novel instances
- Note that a novel instance may have a normalized value outside 0 and 1
  - Why? Is it a big issue?

ARFF Files

- An ARFF (Attribute-Relation File Format) file is an ASCII text file that describes a Machine Learning dataset (or relation).
  - Developed at the University of Waikato (NZ) for use with the Weka machine learning software (http://www.cs.waikato.ac.nz/~ml/weka).
  - We will use the ARFF format for CS 478
- ARFF files have two distinct sections:
  - Metadata information
    - Name of relation (Data Set)
    - List of attributes and domains
  - Data information
    - Actual instances or rows of the relation
- Optional comments may also be included which give information about the Data Set (lines prefixed with %)

Sample ARFF File

```
% 1. Title: Pizza Database
% 2. Sources:
%      (a) Creator: BYU CS 478 Class…
%      (b) Statistics about the features, etc.

@RELATION Pizza

@ATTRIBUTE Weight          CONTINUOUS
@ATTRIBUTE Crust           {Thick, Thin, Stuffed}
@ATTRIBUTE Cheesiness      CONTINUOUS
@ATTRIBUTE Meat            {True, False}
@ATTRIBUTE Quality         {Great, Good, Fair}

@DATA
.9,    Stuffed,    99,    True,    Great
.1,    Thin,       2,     False,   Fair
?,     Thin,       60,    True,    Good
.6,    Thick,      60,    True,    Great
```

- Any column could be the output, but we will assume that the last column(s) is the output
- What would you do to this data before using it with a perceptron and what would the perceptron look like?

ARFF Files

- More details and syntax information for ARFF files can be found at our website
- Data sets that we have already put into the ARFF format can also be found at our website and linked to from the LS content page

http://axon.cs.byu.edu/data/

- You will use a number of these in your simulations throughout the semester – Always read about the task, features, etc, rather than just plugging in the numbers
- You will create your own ARFF files in some projects, and particularly with the group project

Performance Measures

- There are a number of ways to measure the performance of a learning algorithm:
  - Predictive accuracy of the induced model (or error)
  - Size of the induced model
  - Time to compute the induced model
  - etc.
- We will focus here on accuracy
- Fundamental Assumption:

  Future novel instances are drawn from the same/similar distribution as the training instances

Toolkit Training/Testing Alternatives

- Four methods that we will use with our Toolkit:
  - Training set method: The model is evaluated on the same data set that was used for training
  - Static split test set method: Two distinct data sets are made available to the learning algorithm; one for training and one for testing
  - Random split test set method: A single data set is made available to the learning algorithm and the data set is split such that *x*% of the instances are randomly selected for training and the remainder are used for testing, where you supply the value of *x*.
  - *N*-fold cross-validation

Training Set Method

- Procedure
  - Build model from the dataset
  - Compute accuracy on the same dataset
- Simple but least reliable estimate of future performance on unseen data (a rote learner could score 100%!)
- Not used as a performance metric but it is often useful information in understanding how a machine learning model learns
- This is information which you will typically report in your write-ups and then compare it with how the learner does on a test set, etc.

Static Training/Test Set

- Static Split Approach
  - The data owner makes available to the machine learner two distinct datasets:
    - One is used for learning/training (i.e., inducing a model), and
    - One is used exclusively for testing
- Note that this gives you a way to do repeatable tests
- Can be used for challenges (e.g. to see how everyone does on one particular unseen set, etc.)
- Be careful not to overfit the Test Set ("Gold Standard")

Random Training/Test Set Approach

- Random Split Approach
  - The data owner makes available to the machine learner a single dataset
  - The machine learner splits the dataset into a training and a test set, such that:
    - Instances are randomly assigned to either set
    - The distribution of instances (with respect to the target class) is hopefully similar in both sets due to randomizing the data before the split (stratification is even better but not required here)
    - Typically 60% to 90% of instances is used for training and the remainder for testing – the more data there is the more that can be used for training and still get statistically significant test predictions
  - Useful quick estimate for computationally intensive learners
  - Not statistically optimal (high variance, unless lots of data)
    - Could get a luck or unlucky test set
  - Best to do multiple training runs with different splits. Train and test *m* different splits and then average the accuracy over the *m* runs to get a more statistically accurate prediction of generalization accuracy

N-fold Cross-validation

- Use all the data for both training and testing
  - Statistically more reliable
  - All data can be used which is good for small data sets
- Procedure
  - Partition the randomized dataset (call it D) into N equally-sized subsets $S_1, \ldots, S_N$
  - For $k = 1$ to N
    - Let M_k be the model induced from D - S_k
    - Let a_k be the accuracy of M_k on the instances of the test fold S_k
  - Return $(a_1+a_2+\ldots+a_N)/N$

N-fold Cross-validation (cont.)

- The larger N is, the smaller the variance in the final result
- The limit case where $N = |D|$ is known as *leave-one-out* and provides the most reliable estimate. However, it is typically only practical for small instance sets
- Generally, a value of N=10 is considered a reasonable compromise between time complexity and reliability
- Still must chose an actual model to use during execution - how?
  - Could select the one model that was best on its fold?

N-fold Cross-validation (cont.)

- The larger N is, the smaller the variance in the final result
- The limit case where $N = |D|$ is known as *leave-one-out* and provides the most reliable estimate. However, it is typically only practical for small instance sets
- Generally, a value of N=10 is considered a reasonable compromise between time complexity and reliability
- Still must chose an actual model to use during execution - how?
  - Could select the one model that was best on its fold?
  - All data. With any of the above approaches
- Note that CV is just a better way to estimate how well we will do on novel data, rather than a way to do model selection

Perceptron/Regression Project

See Content Section of LS (Learning Suite)

Also briefly review group project proposal part

Your Project Proposals

- Come up with one carefully proposed idea for a possible group machine learning project, that could be done this semester. This proposal should not be more than one page long. It should include a thoughtful first draft proposal of a) description of the project, b) what features the data set would include and c) how and from where would the data set be gathered and labeled. *Give at least one fully specified example of a data set instance based on your proposed features, including a reasonable representation (continuous, nominal, etc.) and value for each feature. The actual values may be fictional at this time. This effort will cause you to consider how plausible the future data gathering and representation might actually be.*
- Examples – Irvine Data Set to get a feel
  - Stick with supervised classification data sets for the most part
- Tasks which interest you
- Too hard vs Too Easy
  - Data can be gathered in a relatively short time
  - Want you to have to battle with the data/features a bit
- Read instructions ASAP on the web site and start thinking