# Applied Machine Learning Lab: Lab 1

Date: 05th Jan 2015

Time: 8:15 am to 10:15 am IST

## Problem 1: Build a Decision Tree classifier to classify tweets

The goal of this lab experiment is to help the student learn how to build and use decision trees for practical problems.

You are provided with a dataset which is a collection of 1500 tweets. The tweets have the word "congress" in them which may refer to the Mobile World Congress, Indian National Congress and so on. The aim of this lab exercise is to disambiguate the same into three classes – Mobile World Congress(3), Indian National Congress(2) and Others(1).

The high level steps to do this are as follows:

1. Read the dataset file using Python, do any format conversions as needed – such as converting this to a CSV file for ease of viewing in a spreadsheet (like MS Excel)
2. The first column is the tweet and the second represents the class.
3. Clean and tokenize the tweets:
    a. Remove RT, hashtags, hyperlinks, tweet handles, etc.
    b. Tokenize the tweets. You can look at nltk word_tokenize for the same.
4. Checkpoint: Decide and report what features (attributes) would be considered for building the decision tree. This should be done latest by 8:45. This is critical for the following steps.
5. Divide the input instances in to training data and test data, where 80% of instances are to be used for training and 20% for testing. The total instances are 1500 and so you should be roughly using 1200 for training and the remaining for testing.
6. Given this dataset as the input where the target variable is one of the three classes, implement the ID3 algorithm that returns a decision tree. You can look at the ID3 implementation from http://www.onlamp.com/lpt/a/6464 and modify it as needed.
7. Classify the test dataset (300 instances), measure the performance. Here the performance is the ratio of the number of instances correctly classified to the total number of instances that constitute the test dataset.
8. Write a function that helps visualizing the tree using indented text. For each new level of the tree add an indent and write the corresponding attribute.

**Deliverables (By 10:15 a.m. Friday):**

1. Source code of all your py modules.
2. Demo of the classification.

**Facebook (By 09:00 p.m. Sunday):**

A report/post that describes or mentions the following:

a) Which attributes have the maximum discriminatory power in concluding the classes?
b) Which attributes don't seem to make a difference?
c) What is the depth of the tree?
d) Performance numbers and your comments on the same. Did you get a good performance (say > 70%) or bad? What are the possible reasons behind the results you obtained?
e) The tree representation as in the slides (or any other comprehensible visualization).

Please zip all the code and mail to the group.

Best wishes from your faculty and seniors ☺