# CS 540-1: Introduction to Artificial Intelligence
# Homework Assignment #1

**Assigned: 9/9**
**Due: 9/21 before class**

## Hand in your homework:

This homework assignment includes written problems and programming in Java. Hand in hardcopy of the requested written parts of the assignment in class. All pages should be stapled together, and should include a cover sheet on top of which includes your name, login, class section, HW #, date, and, if late, how many days late it is. Electronically hand in files containing the Java code that you wrote for the programming part. See course Web page for instructions.

## Late Policy:

All assignments are due **at the beginning of class** on the due date. One (1) day late, defined as a 24-hour period from the deadline (weekday or weekend), will result in 10% of the total points for the assignment deducted. So, for example, if a 100-point assignment is due on a Wednesday 11 a.m, and it is handed in between Wednesday 11 a.m. and Thursday 11 a.m., 10 points will be deducted. Two (2) days late, 25% off; three (3) days late, 50% off. No homework can be turned in more than three (3) days late. Written questions and program submission have the same deadline. A total of two (2) free late days may be used throughout the semester without penalty.

Assignment grading questions must be raised with the instructor within one week after the assignment is returned.

## Collaboration Policy:

You are to complete this assignment individually. However, you are encouraged to discuss the general algorithms and ideas with classmates, TAs, and instructor in order to help you answer the questions. You are also welcome to give each other examples that are not on the assignment in order to demonstrate how to solve problems. But we require you to:

- not explicitly tell each other the answers
- not to copy answers or code fragments from anyone or anywhere
- not to allow your answers to be copied
- not to get any code on the Web

In those cases where you work with one or more other people on the general discussion of the assignment and surrounding topics, we suggest that you specifically record on the assignment the names of the people you were in discussion with.

**Question 1**: Warm Up Math Questions [10]

a) [4] There are 5 tokens in a bag. They have the same shape but different colors and are worth 1,2,3,4,5 dollars, respectively. Without watching, you randomly grab 2 tokens from the bag. What is the chance that together the two tokens are worth 6 dollars or more?
b) [3] Take the derivative with respect to $x$: $(\ln x + 3x)^2 + xe^x$
c) [3] Compute $\lim_{x \to 0^+} x \ln x$. Show your steps.

**Question 2**: k-means Clustering [25]

A. [10] Given a dataset with five points (0,0),(3,3),(5,2),(6,3),(7,0), k=2 clusters, and initial centers $c_1$=(4, 2), $c_2$=(6, 1) , run k-means clustering by hand (you can use a calculator). Use Euclidean distance. For <u>each iteration</u>, show
   a) The cluster membership for each point, under the current cluster centers
   b) The distortion, under the current cluster centers
   c) The new updated cluster centers
B. [10] Repeat A, but with initial centers at c1=(-1,1), c2=(0, 5).
C. [5] Briefly discuss what property of k-mean A and B shows.

**Question 3**: kNN Classification [25]

Given a dataset with binary labels (x,y)={(1,+), (2, -), (4,+), (7,+), (11,-)}, compute the

a) [4] kNN <u>training set error</u> with k=5, using 0-1 loss and Euclidean distance. If there is a tie, always favor the positive (+) class.
b) [4] Same as above but with k=4
c) [4] Same as above but with k=3
d) [4] Same as above but with k=2
e) [4] Same as above but with k=1
f) [5] Should we choose the k with the smallest <u>training set error</u>? Why?

**Question 4: Programming** : Animals in Our Mind [40]

In this question, you will program hierarchical clustering in JAVA to cluster animals. The data we use comes from a Dutch psychology study by De Deyne

*et al*. First, take a look at `features.txt`, which has 764 lines. Each line is the name of a feature (the names are translated to English, some may look funny). Second, take a look at `animals.txt`, which has 25 animal names. These two files are for your information and help you debug, but your program will not need them.

The actual feature file is `matrix.txt`, which has 764 rows and 25 columns. Each column is for an animal, in the order specified in *animals.txt*. Each row is for a feature, as specified in *features.txt*. The entries are integers from 0 to 4. The experiment asked questions like "Does animal X have feature Y?" to four human participants. The entry at column X and row Y specifies the number of participants who answered "Yes" to that question. For example, the entry at column 1 row 1 is 2, meaning 2 out of 4 participants thought `squirrel` have the feature `appears_in_comics`.

(Important: we will test your program on a different matrix with potentially different features and animals. Do not hard-code any numbers above. Please see the command-line format below. You may assume that `matrix.txt` is always a well-formed plain text file containing an integer array.)

Use the columns as feature vectors. Cluster the animals using hierarchical clustering. Specifically,

1. Name your program HW1.java
2. Use complete-linkage.
3. Use Euclidean distance.
4. Represent each animal by its column ID: the first column (in this case squirrel) has ID 1, the second column (deer) has ID 2, and so on. This way, your program doesn't need to handle actual animal names in `animals.txt`, and your task is simplified a bit.
5. In case of a tie on which two clusters to merge next, prefer the cluster which contains the smallest animal ID.
6. Stop when the specified number of clusters is achieved (see command-line format below).

**Command-line format:**

Your program `HW1.java` should be able to run on the department's Linux machines with the following command:

```
HW1 nAnimals nFeatures datafile nClusters
```

`nAnimals`: the number of columns (such as 25)
`nFeatures`: the number of rows (such as 764)
`datafile`: the name of the data file, such as `matrix.txt`
`nClusters`: the number of desired clusters

**Output format:**
Your program should print out `nClusters` lines. Each line is for a cluster, with the IDs in that cluster sorted from small to large, and separated by whitespace. The clusters should be sorted by the smallest ID in them. For example, say we have 6 animals and 3 clusters, a possible output might be:

```
1 3 6
2
4 5
```

Do not print out anything else (this helps our grading program).

**Deliverables:**

1. Handin all relevant ".java" and ".class" files needed for your program.
2. Handin a Makefile to compile the code.
3. Optionally, in the written part of your homework, add any comments about the program that you would like us to know.

Hand in your code as specified by the course webpage. Our SECTION is 1, and the ASSISGNMENT_NAME is HW1.