1. (Support Vectors) For an SVM, if we remove one of the support vectors from the training set, does the size of the maximum margin decrease, stay the same, or increase for that dataset? Why? Also justify your answer by providing a simple dataset (no more than 2-D) in which you identify the support vectors, draw the location of the maximum margin hyperplane, remove one of the support vectors, and draw the location of the resulting maximum margin hyperplane.

2. (Cubic Kernels) We showed in class that the quadratic kernel $K(x_i, x_j) = (x_i x_j)^2$ was equivalent to mapping each $x$ into a higher dimensional space where

$$\Phi(x) = (x_1^2, x_2^2, \sqrt{2}x_1x_2)$$

for the case where $x = (x_1, x_2)$. Now consider the cubic kernel $K(x_i, x_j) = (x_i, x_j + 1)^3$. Note that this kernel adds 1 to the dot product. What is the corresponding $\Phi$ function, again for the case where $x = (x_1, x_2)$?

3. (Non-linear kernels) Consider the 2-bit XOR problem for which the entire instance space is as follows:

<table>
<thead>
<tr>
<th>$y$</th>
<th>$x_1$</th>
<th>$x_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>+1</td>
<td>-1</td>
<td>1</td>
</tr>
<tr>
<td>+1</td>
<td>1</td>
<td>-1</td>
</tr>
<tr>
<td>-1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

These instances are not linearly separable, but they are separable with a polynomial kernel. Recall that the polynomial kernel is of the form $K(x_i, x_j) = (x_i x_j + c)^d$ where $c$ and $d$ are integers. Select values for $c$ and $d$ that yield a space in which the instances above are linearly separable. Write down the mapping $\Phi$ to which this kernel corresponds, write down $\Phi(x)$ for each instance above, and write down the parameters of any hyperplane in the expanded space that perfectly classifies the instances.

4. (SVM tools) In this exercise you will get some experience with a state-of-the-art SVM package, and will compare the performance of various kernels. You will need to do the following:

- Download SVMlight from http://www.cs.cornell.edu/People/tj/svm_light/. This web page has links to source code and binaries, and contains information on how to build and run the various SVMlight tools.
- Download the ionosphere dataset from the course web page. This dataset comes from the UC Irvine Machine Learning Repository. The file ionosphere.names describes the contents of the dataset. The file ionosphere.data contains the actual data in a format that is accepted by SVMlight. You’ll need to manually divide the data into training and testing sets. One split will suffice.
- Are the data linearly separable? Describe how you used SVMlight to determine this.
- Compare the performance of the SVM using each of the following kernels - linear, polynomial, radial basis. The polynomial and radial basis kernels each take parameters. Experiment with various values of these parameters and $C$, which controls the tradeoff between training error and margin. Write a short report detailing your experiments. Be sure to report the margin and number of support vectors for various settings of the parameters. What kernel performed the best? What does this say, if anything, about the data?
5. (Bayes nets) This question asks you to consider the construction and use of Bayesian network in an automated suggestion system inside a word processor. The basic idea is to develop a (very simple) probabilistic system for user modelling that will help us to assess when a user might benefit from suggestions, and to decide when to make suggestions. We focus only on the probabilistic estimation of the user model, not on the decision making. (But it should be clear how one would use this probabilistic information to make decisions about whether to offer suggestions.) We have built into our word processor an adaptive system that will help a user by suggesting words as they type. Basically, our system has the ability to suggest the most likely word the user is attempting to type. If (a) the user pauses or backspaces in the middle of typing a word, and (b) the system estimates that the word is difficult to spell, then the system can pop up a suggestion box with its best guess about the word the user is trying to type. The user can then mouse over the suggestion box and dismiss or accept the suggestion. In initial tests of our technology, we have found that some users like suggestions constantly being popped up, and others do not. Preliminary studies suggest that each user has a certain degree of independence that makes them more or less likely to even consider suggestions. Basically, more independent users are less likely to consider suggestions. Any suggestion we make that goes unconsidered is likely to annoy the user (and independent users are more likely to be annoyed).

As a precursor to making our system more adaptive, we wish to develop a Bayesian network that will help assess a specific users degree of independence. Your job is to build a small version of this network. We imagine that the system makes three consecutive suggestions. Based on the observed user behavior, we want to assess their degree of independence and whether or not any specific suggestion was annoying to them. We use the following variables:

- \( I \in \{v, m, n\} \): \( I \) stands for the user’s level of independence. \( v \) means “very independent”, \( m \) means “moderately independent”, and \( n \) means “not independent”.
- \( M_1 \in \{yes, no\} \): did the user “mouse over” the first suggestion box. We take a mouse over to mean that the user considered the suggestion the system made.
- \( M_2, M_3 \): these are analogous to \( M_1 \), referring to the second and third suggestions the system made.
- \( A_1 \in \{yes, no\} \): was the user “annoyed” by the first suggestion.
- \( A_2, A_3 \): these are analogous to \( A_1 \), referring to the second and third suggestions.

We have the following probabilistic information about these variables (from which you should be able to infer the appropriate conditional independence relationships):

- Preliminary studies suggest that 30% of users are very independent, 50% are moderately independent, and 20% are not independent.
- Very independent users consider suggestions with probability 0.1; moderately independent users consider suggestions with probability 0.3; users that are not independent consider suggestions with probability 0.7.
- The probability with which a user is annoyed by a suggestion depends on whether they considered it. Very independent users have a 90% chance of finding a suggestion box annoying if they don’t consider the suggestion, but only a 40% chance of being annoyed if they do consider the suggestion. Moderately independent users have an 80% chance of being annoyed by unconsidered suggestions, and a 20% chance of being annoyed by suggestions they consider. And users that are not independent find unconsidered suggestions annoying with probability 0.5, and considered suggestions annoying with probability 0.1.

**Part 1:** Draw a Bayesian network over the variables described. For each variable, show its corresponding CPT. The arcs defining your Bayes Network should accurately capture the probabilistic dependencies between these variables.

**Part 2:** With respect to your Bayesian network from part 1, answer the following questions. Give a justification for your answer using the d-separation criterion: if d-separation holds, state why each undirected path is blocked; if it does not hold, describe one undirected path which is unblocked.
• Are $M_1$ and $M_3$ independent?
• Are $M_1$ and $M_3$ independent given $I$?
• Are $A_1$ and $A_2$ independent?
• Are $A_1$ and $M_3$ independent given $M_1$?