ANDREW ID (CAPITALS): ____________________________

NAME (CAPITALS): ____________________________

10-701/15-781 Final, Fall 2003

- You have 3 hours.
- There are 10 questions. If you get stuck on one question, move on to others and come back to the difficult question later.
- The maximum possible total score is 100.
- Unless otherwise stated there is no need to show your working.
- Good luck!

1 Short Questions (16 points)

(a) Traditionally, when we have a real-valued input attribute during decision-tree learning we consider a binary split according to whether the attribute is above or below some threshold. Pat suggests that instead we should just have a multiway split with one branch for each of the distinct values of the attribute. From the list below choose the single biggest problem with Pat's suggestion:

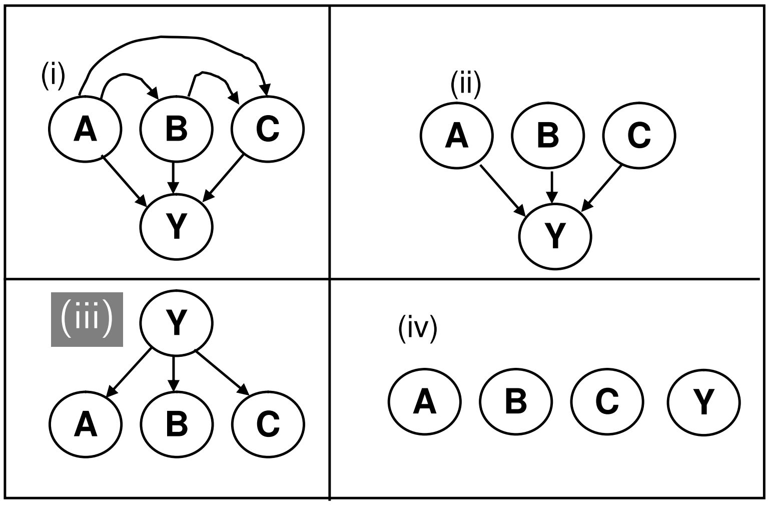
(i) It is too computationally expensive.

(ii) It would probably result in a decision tree that scores badly on the training set and a testset.

[✓ (iii)] It would probably result in a decision tree that scores well on the training set but badly on a testset.

(iv) It would probably result in a decision tree that scores well on a testset but badly on a training set.

(b) You have a dataset with three categorical input attributes A, B and C. There is one categorical output attribute Y. You are trying to learn a Naive Bayes Classifier for predicting Y. Which of these Bayes Net diagrams represents the naive bayes classifier assumption?



(c) For a neural network, which one of these structural assumptions is the one that most affects the trade-off between underfitting (i.e. a high bias model) and overfitting (i.e. a high variance model):

[✓ (i)] The number of hidden nodes

(ii) The learning rate

(iii) The initial choice of weights

(iv) The use of a constant-term unit input

(d) For polynomial regression, which one of these structural assumptions is the one that most affects the trade-off between underfitting and overfitting:

[✓ (i)] The polynomial degree

(ii) Whether we learn the weights by matrix inversion or gradient descent

(iii) The assumed variance of the Gaussian noise

(iv) The use of a constant-term unit input

(e) For a Gaussian Bayes classifier, which one of these structural assumptions is the one that most affects the trade-off between underfitting and overfitting:

(i) Whether we learn the class centers by Maximum Likelihood or Gradient Descent

[✓ (ii)] Whether we assume full class covariance matrices or diagonal class covariance matrices

(iii) Whether we have equal class priors or priors estimated from the data.

(iv) Whether we allow classes to have different mean vectors or we force them to share the same mean vector

(f) For Kernel Regression, which one of these structural assumptions is the one that most affects the trade-off between underfitting and overfitting:

(i) Whether kernel function is Gaussian versus triangular versus box-shaped

(ii) Whether we use Euclidian versus L_1 versus L_∞ metrics

[✓ (iii)] The kernel width

(iv) The maximum height of the kernel function

(g) **(True or** [✓ **False**]**)** Given two classifiers A and B, if A has a lower VC-dimension than B then A almost certainly will perform better on a testset.

(h) $P(\textit{Good Movie} \mid \textit{Includes Tom Cruise}) = 0.01$
$P(\textit{Good Movie} \mid \textit{Tom Cruise absent}) = 0.1$
$P(\textit{Tom Cruise in a randomly chosen movie}) = 0.01$

What is $P(\textit{Tom Cruise is in the movie} \mid \textit{Not a Good Movie})$?

$$T \sim \textit{Tom Cruise is in the movie}$$
$$G \sim \textit{Good Movie}$$

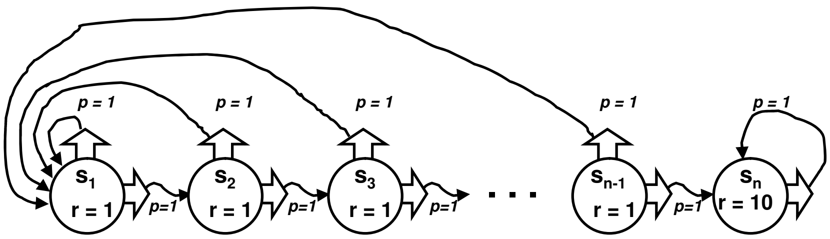
$$P(T|\tilde{}G) = \frac{P(T, \tilde{}G)}{P(\tilde{}G)}$$
$$= \frac{P(\tilde{}G|T)P(T)}{P(\tilde{}G|T)P(T) + P(\tilde{}G|\tilde{}T)P(\tilde{}T)}$$
$$= \frac{0.01 \times (1 - 0.01)}{0.01 \times (1 - 0.01) + (1 - 0.1) \times (1 - 0.01)}$$
$$= 1/91 \approx 0.01099$$

2 Markov Decision Processes (13 points)

For this question it might be helpful to recall the following geometric identities, which assume $0 \leq \alpha < 1$.

$$\sum_{i=0}^{k} \alpha^i = \frac{1 - \alpha^{k+1}}{1 - \alpha} \qquad\qquad \sum_{i=0}^{\infty} \alpha^i = \frac{1}{1 - \alpha}$$

The following figure shows an MDP with N states. All states have two actions (North and Right) except S_n, which can only self-loop. Unlike most MDPs, all state transitions are deterministic. Assume discount factor γ.



For questions (a)–(e), express your answer as a finite expression (no summation signs or . . . 's) in terms of n and/or γ.

(a) What is $J^*(S_n)$?

$$J^*(S_n) = 10 + \gamma \cdot J^*(S_n) \Longrightarrow J^*(S_n) = \frac{10}{1 - \gamma}$$

(b) There is a unique optimal policy. What is it?

$A_i =$ Right $(i = 1, \ldots, n)$

(c) What is $J^*(S_1)$?

$$J^*(S_1) = 1 + \gamma + \cdots + \gamma^{n-2} + J^*(S_n) \cdot \gamma^{n-1} = \frac{1 + 9\gamma^{n-1}}{1 - \gamma}$$

(d) Suppose you try to solve this MDP using value iteration. What is $J^1(S_1)$?

$J^1(S_1) = 1$

(e) Suppose you try to solve this MDP using value iteration. What is $J^2(S_1)$?

$J^2(S_1) = 1 + \gamma$

(f) Suppose your computer has exact arithmetic (no rounding errors). How many iterations of value iteration will be needed before all states record their exact (correct to infinite decimal places) J^* value? Pick one:

(i) Less than $2n$

(ii) Between $2n$ and n^2

(iii) Between $n^2 + 1$ and 2^n

[✓ (iv)] It will never happen

It's a limiting process.

(g) Suppose you run policy iteration. During one step of policy iteration you compute the value of the current policy by computing the exact solution to the appropriate system of n equations in n unknowns. Suppose too that when choosing the action during the policy improvement step, ties are broken by choosing North.

Suppose policy iteration begins with all states choosing North.

How many steps of policy iteration will be needed before all states record their exact (correct to infinite decimal places) J^* value? Pick one:

[✓ (i)] Less than $2n$

(ii) Between $2n$ and n^2

(iii) Between $n^2 + 1$ and 2^n

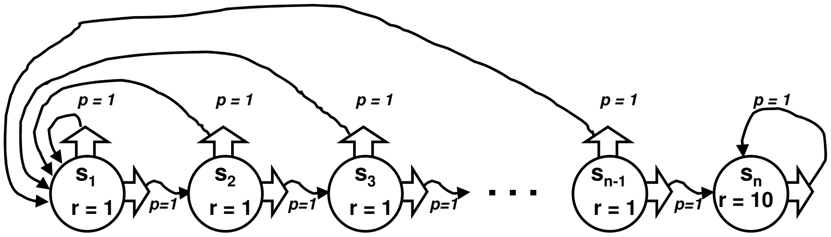
(iv) It will never happen

After i policy iterations, we have

$$Action(S_j) = \begin{cases} Right & \text{if } n - i < j < n \\ North & \text{otherwise.} \end{cases}$$

3 Reinforcement Learning (10 points)

This question uses the same MDP as the previous question, repeated here for your convenience. Again, assume $\gamma = \frac{1}{2}$.



Suppose we are discovering the optimal policy via Q-learning. We begin with a Q-table initialized with 0's everywhere:

$Q(S_i, North) = 0$ for all i
$Q(S_i, Right) = 0$ for all i

Because the MDP is determistic, we run Q-learning with a learning rate $\alpha = 1$. Assume we start Q-learning at state S_1.

(a) Suppose our exploration policy is to always choose a random action. How many steps do we expect to take before we first enter state S_n?

(i) $O(n)$ steps
(ii) $O(n^2)$ steps
(iii) $O(n^3)$ steps
[✓ (iv)] $O(2^n)$ steps
(v) It will certainly never happen

You are expected to visit S_i twice before entering S_{i+1}.

(b) Suppose our exploration is greedy and we break ties by going North:

Choose North if $Q(S_i, North) \geq Q(S_i, Right)$
Choose Right if $Q(S_i, North) < Q(S_i, Right)$

How many steps do we expect to take before we first enter state S_n?

(i) $O(n)$ steps
(ii) $O(n^2)$ steps
(iii) $O(n^3)$ steps
(iv) $O(2^n)$ steps
[✓ (v)] It will certainly never happen

The exploration sequence is $S_1 S_1 S_1 \ldots$

(c) Suppose our exploration is greedy and we break ties by going Right:

Choose North if $Q(S_i, North) > Q(S_i, Right)$
Choose Right if $Q(S_i, North) \leq Q(S_i, Right)$

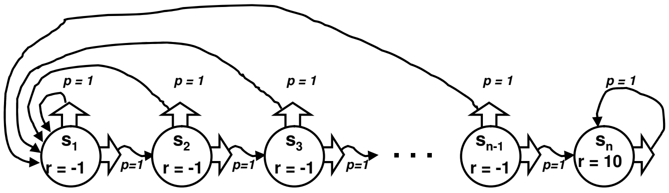
How many steps do we expect to take before we first enter state S_n?

[✓ (i)] $O(n)$ steps
(ii) $O(n^2)$ steps
(iii) $O(n^3)$ steps
(iv) $O(2^n)$ steps
(v) It will certainly never happen

The exploration sequence is $S_1 S_2 S_3 \ldots S_{n-1} S_n$.

WARNING: Question (d) is only worth 1 point so you should probably just guess the answer unless you have plenty of time.

(d) In this question we work with a similar MDP except that each state other than S_n has a punishment (-1) instead of a reward (+1). S_n remains the same large reward (10). The new MDP is shown below:



Suppose our exploration is greedy and we break ties by going North:

Choose North if $Q(S_i, North) \geq Q(S_i, Right)$
Choose Right if $Q(S_i, North) < Q(S_i, Right)$

How many steps do we expect to take before we first enter state S_n?

(i) $O(n)$ steps
(ii) $O(n^2)$ steps
(iii) $O(n^3)$ steps
(iv) $O(2^n)$ steps
(v) It will certainly never happen

(ii) or (iii).

Each time a new state S_i is visited, we have to go North and jump back to S_1. So the sequence should be longer than $S_1 S_{1:2} S_{1:3} \ldots S_{1:n}$, i.e. it takes at least $O(n^2)$ steps.

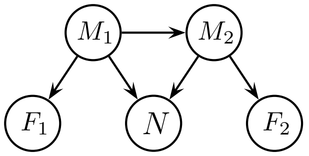
The jump from S_j to S_1 happens more than once because $Q(S_j, Right)$ keeps increasing. But the sequence should be shorter than $\{S_1\}\{S_1 S_{1:2}\}\{S_1 S_{1:2} S_{1:3}\} \ldots \{S_1 S_{1:2} \cdots S_{1:n}\}$, i.e. it takes at most $O(n^3)$ steps.

4 Bayesian Networks (11 points)

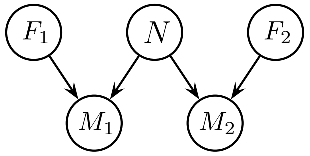
Construction. Two astronomers in two different parts of the world, make measurements M_1 and M_2 of the number of stars N in some small regions of the sky, using their telescopes. Normally, there is a small possibility of error by up to one star in each direction. Each telescope can be, with a much smaller probability, badly out of focus (events F_1 and F_2). In such a case the scientist will undercount by three or more stars or, if N is less than three, fail to detect any stars at all.

For questions (a) and (b), consider the four networks shown below.

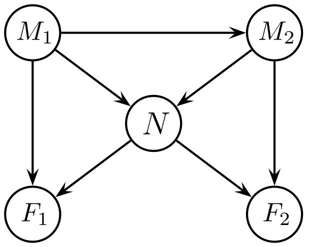
(i)



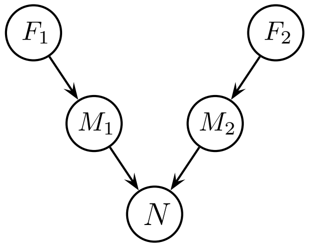
(ii)



(iii)



(iv)



(a) Which of them correctly, but not necessarily efficiently, represents the above information? **Note that there may be multiple answers**.

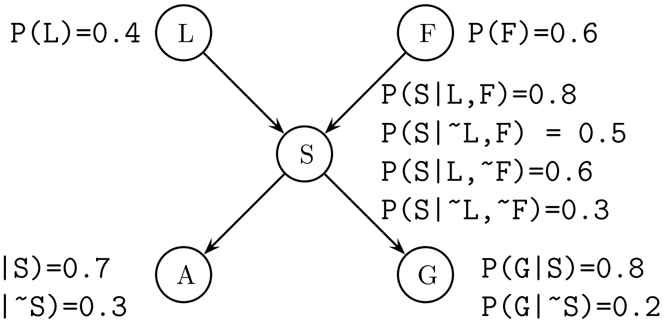
(ii) and (iii).

(ii) can be constructed directly from the physical model. (iii) is equivalent to (ii) with a different ordering of variables. (i) is incorrect because F_i and N cannot be conditionally independent given M_i. (iv) is incorrect because M_1 and M_2 cannot be independent.

(b) Which is the best network?

(ii). Intuitive and easy to interpret. Less links thus less CPT entries. Easier to assign the values of CPT entries.

Inference. A student of the Machine Learning class notices that people driving SUVs (S) consume large amounts of gas (G) and are involved in more accidents than the national average (A). He also noticed that there are two types of people that drive SUVs: people from Pennsylvania (L) and people with large families (F). After collecting some statistics, he arrives at the following Bayesian network.



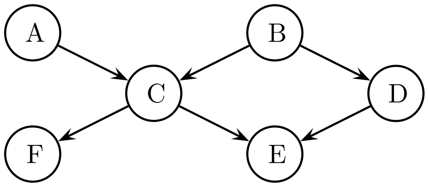
(c) What is $P(S)$?

$$\begin{aligned} P(S) &= P(S|L,F)P(L)P(F) + P(S|{}^\sim L,F)P({}^\sim L)P(F) + \\ &\quad P(S|L,{}^\sim F)P(L)P({}^\sim F) + P(S|{}^\sim L,{}^\sim F)P({}^\sim L)P({}^\sim F) \\ &= 0.4 \cdot 0.6 \cdot 0.8 + 0.6 \cdot 0.6 \cdot 0.5 + 0.4 \cdot 0.4 \cdot 0.6 + 0.6 \cdot 0.4 \cdot 0.3 \\ &= 0.54 \end{aligned}$$

(d) What is $P(S|A)$?

$$P(S|A) = \frac{P(S,A)}{P(A|S)P(S) + P(A|{}^\sim S)P({}^\sim S)} = \frac{0.54 \cdot 0.7}{0.54 \cdot 0.7 + 0.46 \cdot 0.3} = 0.733$$

Consider the following Bayesian network. State whether the given conditional independences are implied by the net structure.



(f) **([✓ True] or False)** `I<A,{},B>`

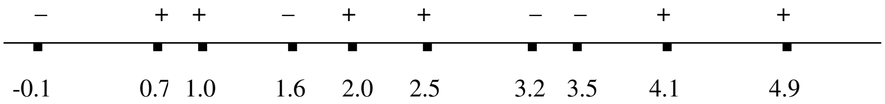
(g) **(True or [✓ False])** `I<A,{E},D>`

(h) **(True or [✓ False])** `I<A,{F},D>`

5 Instance Based Learning (8 points)

Consider the following dataset with one real-valued input x and one binary output y. We are going to use k-NN with unweighted Euclidean distance to predict y for x.

X	Y
-0.1	-
0.7	+
1.0	+
1.6	-
2.0	+
2.5	+
3.2	-
3.5	-
4.1	+
4.9	+



(a) What is the leave-one-out cross-validation error of 1-NN on this dataset? Give your answer as the number of misclassifications.

4

(b) What is the leave-one-out cross-validation error of 3-NN on this dataset? Give your answer as the number of misclassifications.

8

Consider a dataset with N examples: $\{(x_i, y_i)|1 \leq i \leq N\}$, where both x_i and y_i are real valued for all i. Examples are generated by $y_i = w_0 + w_1 x_i + e_i$ where e_i is a Gaussian random variable with mean 0 and standard deviation 1.

(c) We use least square linear regression to solve w_0 and w_1, that is

$\{w_0^*, w_1^*\} = \arg \min_{\{w_0, w_1\}} \sum_{i=1}^{N} (y_i - w_0 - w_1 x_i)^2$.

We assume the solution is unique. Which one of the following statements is true?

(i) $\sum_{i=1}^{N} (y_i - w_0^* - w_1^* x_i) y_i = 0$

(ii) $\sum_{i=1}^{N} (y_i - w_0^* - w_1^* x_i) x_i^2 = 0$

[✓ (iii)] $\sum_{i=1}^{N} (y_i - w_0^* - w_1^* x_i) x_i = 0$

(iv) $\sum_{i=1}^{N} (y_i - w_0^* - w_1^* x_i)^2 = 0$

(d) We change the optimization criterion to include local weights, that is

$\{w_0^*, w_1^*\} = \arg \min_{\{w_0, w_1\}} \sum_{i=1}^{N} \alpha_i^2 (y_i - w_0 - w_1 x_i)^2$

where α_i is a local weight. Which one of the following statements is true?

(i) $\sum_{i=1}^{N} \alpha_i^2 (y_i - w_0^* - w_1^* x_i)(x_i + \alpha_i) = 0$

(ii) $\sum_{i=1}^{N} \alpha_i (y_i - w_0^* - w_1^* x_i) x_i = 0$

(iii) $\sum_{i=1}^{N} \alpha_i^2 (y_i - w_0^* - w_1^* x_i)(x_i y_i + w_1^*) = 0$

[✓ (iv)] $\sum_{i=1}^{N} \alpha_i^2 (y_i - w_0^* - w_1^* x_i) x_i = 0$

6 VC-dimension (9 points)

Let H denote a hypothesis class, and $VC(H)$ denote its VC dimension.

(a) **(True or [✓ False])** If there exists a set of k instances that *cannot* be shattered by H, then $VC(H) < k$.

(b) **([✓ True] or False)** If two hypothesis classes H_1 and H_2 satisfy $H_1 \subseteq H_2$, then $VC(H_1) \leq VC(H_2)$.

(c) **(True or [✓ False])** If three hypothesis classes H_1, H_2 and H_3 satisfy $H_1 = H_2 \cup H_3$, then $VC(H_1) \leq VC(H_2) + VC(H_3)$.

A counter example:

$H_2 = \{h\}, h = 0$ and $H_3 = \{h'\}, h' = 1$. Apparently $VC(H_2) = VC(H_3) = 0$.

$H_1 = H_2 \cup H_3 = \{h, h'\}$.

So $VC(H_1) = 1 > VC(H_2) + VC(H_3) = 0$.

For questions (d)–(f), give $VC(H)$. No explanation is required.

(d) $H = \{h_\alpha | 0 \leq \alpha \leq 1, h_\alpha(x) = 1 \text{ iff } x \geq \alpha \text{ otherwise } h_\alpha(x) = 0\}$.

1

(e) H is the set of all perceptrons in 2D plane, i.e.
$H = \{h_{\mathbf{w}} | h_{\mathbf{w}} = \theta(w_0 + w_1 x_1 + w_2 x_2) \text{ where } \theta(z) = 1 \text{ iff } z \geq 0 \text{ otherwise } \theta_z = 0\}$.

3

(f) H is the set of all circles in 2D plane. Points inside the circles are classified as 1 otherwise 0.

3

7 SVM and Kernel Methods (8 points)

(a) Kernel functions implicitly define some mapping function $\phi(\cdot)$ that transforms an input instance $\mathbf{x} \in \mathbb{R}^d$ to a high dimensional feature space Q by giving the form of dot product in Q: $K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)$.

Assume we use radial basis kernel function $K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\frac{1}{2}\|\mathbf{x}_i - \mathbf{x}_j\|^2)$. Thus we assume that there's some implicit unknown function $\phi(\mathbf{x})$ such that

$$\phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j) = K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\frac{1}{2}\|\mathbf{x}_i - \mathbf{x}_j\|^2)$$

Prove that for any two input instances $\mathbf{x}_i$ and $\mathbf{x}_j$, the squared Euclidean distance of their corresponding points in the feature space Q is less than 2, i.e. prove that $\|\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)\|^2 < 2$.

$$\begin{aligned}
&\|\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)\|^2 \\
=&(\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)) \cdot (\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)) \\
=&\phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_i) + \phi(\mathbf{x}_j) \cdot \phi(\mathbf{x}_j) - 2 \cdot \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j) \\
=&2 - 2\exp(-\frac{1}{2}\|\mathbf{x}_i - \mathbf{x}_j\|^2) \\
<&2
\end{aligned}$$

(b) With the help of a kernel function, SVM attempts to construct a hyper-plane in the feature space Q that maximizes the margin between two classes. The classification decision of any $\mathbf{x}$ is made on the basis of the sign of

$$\hat{\mathbf{w}}^T\phi(\mathbf{x}) + \hat{w}_0 = \sum_{i \in SV} y_i \alpha_i K(\mathbf{x}_i, \mathbf{x}) + \hat{w}_0 = f(\mathbf{x}; \alpha, \hat{w}_0),$$

where $\hat{\mathbf{w}}$ and $\hat{w}_0$ are parameters for the classification hyper-plane in the feature space Q, SV is the set of support vectors, and α_i is the coefficient for the support vector.

Again we use the radial basis kernel function. Assume that the training instances are linearly separable in the feature space Q, and assume that the SVM finds a margin that perfectly separates the points.

(True or False) If we choose a test point $\mathbf{x}_{far}$ which is far away from any training instance $\mathbf{x}_i$ (distance here is measured in the original space $\mathbb{R}^d$), we will observe that $f(\mathbf{x}_{far}; \alpha, \hat{w}_0) \approx \hat{w}_0$.

$$\begin{aligned}
&\|\mathbf{x}_{far} - \mathbf{x}_i\| \gg 0,\ \forall i \in SV \\
\Longrightarrow &K(\mathbf{x}_{far}, \mathbf{x}_i) \approx 0,\ \forall i \in SV \\
\Longrightarrow &\textstyle\sum_{i \in SV} y_i \alpha_i K(\mathbf{x}_{far}, \mathbf{x}_i) \approx 0 \\
\Longrightarrow &f(\mathbf{x}_{far}; \alpha, \hat{w}_0) \approx \hat{w}_0
\end{aligned}$$

(c) **(True or False)** The SVM learning algorithm is guaranteed to find the globally optimal hypothesis with respect to its object function.

See Burges' tutorial.

(d) **(True or False)** The VC dimension of a Perceptron is smaller than the VC dimension of a simple linear SVM.

Both Perceptron and linear SVM are linear discriminators (i.e. a line in 2D space or a plane in 3D space . . .), so they should have the same VC dimension.

(e) **(True or False)** After being mapped into feature space Q through a radial basis kernel function, a Perceptron may be able to achieve better classification performance than in its original space (though we can't guarantee this).

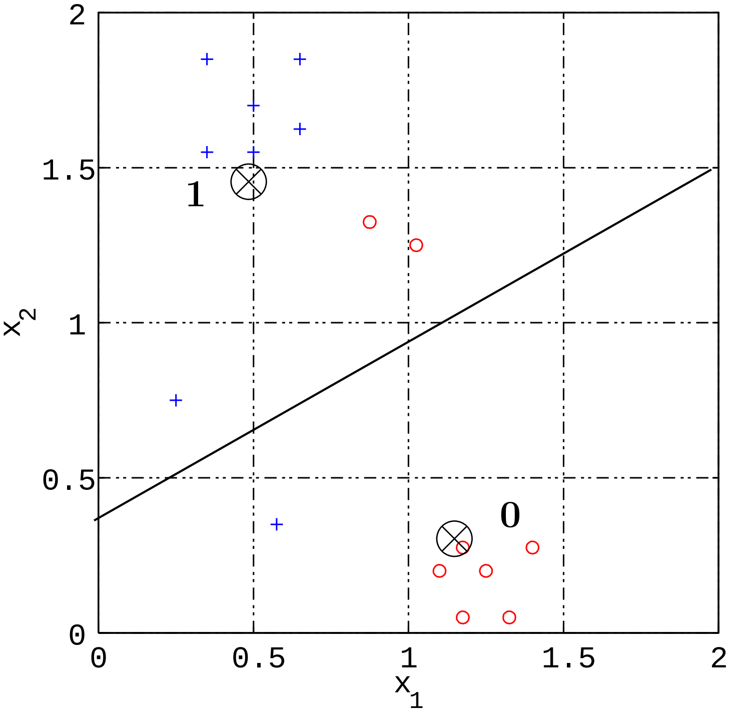
Sometimes it isn't sufficient for a given learning algorithm to work in the input space because the assumption behind the algorithm doesn't match the real pattern of the data. For example, SVM and Perceptron require the data are linearly separable. When the assumption isn't held, we may apply some kind of transformation to the data, mapping them to a new space where the learning algorithm can be used. Kernel function provides us a means to define the transformation. You may have read some papers that report improvements on classification performance using kernel function. However, the improvements are usually obtained from careful selection and tuning of parameters. Namely, we can't guarantee the improvements are always available.

(f) **(True or False)** After mapped into feature space Q through a radial basis kernel function, 1-NN using unweighted Euclidean distance may be able to achieve better classification performance than in original space (though we can't guarantee this).

Suppose $\mathbf{x}_i$ and $\mathbf{x}_j$ are two neighbors for the test instance $\mathbf{x}$ such that $\|\mathbf{x}-\mathbf{x}_i\| < \|\mathbf{x}-\mathbf{x}_j\|$. After mapped to feature space, $\|\phi(\mathbf{x}) - \phi(\mathbf{x}_i)\|^2 = 2 - 2\exp(-\frac{1}{2}\|\mathbf{x} - \mathbf{x}_i\|^2) < 2 - 2\exp(-\frac{1}{2}\|\mathbf{x} - \mathbf{x}_j\|^2) = \|\phi(\mathbf{x}) - \phi(\mathbf{x}_j)\|^2$. So, if $\mathbf{x}_i$ is the nearest neighbor of $\mathbf{x}$ in the original space, it will also be the nearest neighbor in the feature space. Therefore, 1-NN doesn't work better in the feature space. Please note that k-NN using non-Euclidean distance or weighted voting may work.

8 GMM (8 points)

Consider the classification problem illustrated in the following figure. The data points in the figure are labeled, where "o" corresponds to class 0 and "+" corresponds to class 1. We now estimate a GMM consisting of 2 Gaussians, one Gaussian per class, with the constraint that the covariance matrices are identity matrices. The mixing proportions (class frequencies) and the means of the two Gaussians are free parameters.



(a) Plot the maximum likelihood estimates of the means of the two Gaussians in the figure. Mark the means as points "x" and label them "0" and "1" according to the class.

The means of the two Gaussians should be close to the center of mass of the points.

(b) Based on the learned GMM, what is the probability of generating a new data point that belongs to class 0?

0.5

(c) How many data points are classified *incorrectly*?

3

(d) Draw the decision boundary in the same figure.

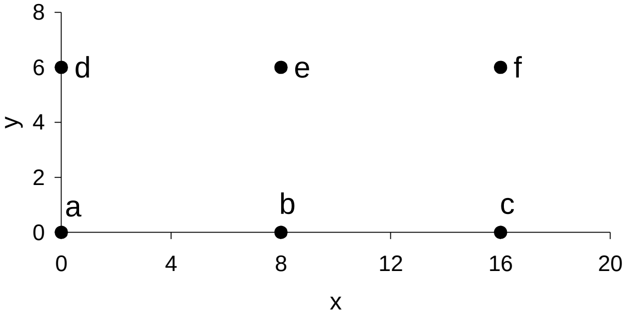
Since the two classes have the same number of points and identical covariance matrices, the decision boundary should be a straight line, which is also the orthogonal bisector of the line segment connecting the class means.

9 K-means Clustering (9 points)

There is a set S consisting of 6 points in the plane shown as below, $a = (0, 0)$, $b = (8, 0)$, $c = (16, 0)$, $d = (0, 6)$, $e = (8, 6)$, $f = (16, 6)$. Now we run the k-means algorithm on those points with $k = 3$. The algorithm uses the Euclidean distance metric (i.e. the straight line distance between two points) to assign each point to its nearest centroid. Ties are broken in favor of the centroid to the left/down. Two definitions:

- A **k-starting configuration** is a subset of k starting points from S that form the initial centroids, e.g. $\{a, b, c\}$.
- A **k-partition** is a partition of S into k non-empty subsets, e.g. $\{a, b, e\}, \{c, d\}, \{f\}$ is a 3-partition.

Clearly any k-partition induces a set of k centroids in the natural manner. A k-partition is called *stable* if a repetition of the k-means iteration with the induced centroids leaves it unchanged.



(a) How many 3-starting configurations are there? (Remember, a 3-starting configuration is just a subset, of size 3, of the six datapoints).

$C_6^3 = 20$

(b) Fill in the following table:

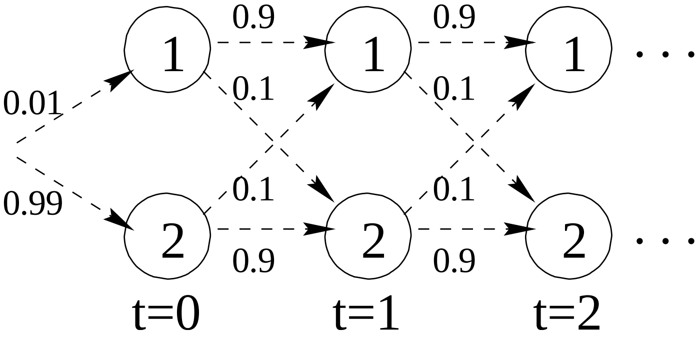
3-partition	Stable?	An example 3-starting configuration that can arrive at the 3-partition after 0 or more iterations of k-means (or write "none" if no such 3-starting configuration exists)	# of unique 3-starting configurations that arrive at the 3-partition
$\{a, b, e\}, \{c, d\}, \{f\}$	N	none	0
$\{a, b\}, \{d, e\}, \{c, f\}$	Y	{b, c, e}	4
$\{a, d\}, \{b, e\}, \{c, f\}$	Y	{a, b, c}	8
$\{a\}, \{d\}, \{b, c, e, f\}$	Y	{a, b, d}	2
$\{a, b\}, \{d\}, \{c, e, f\}$	Y	none	0
$\{a, b, d\}, \{c\}, \{e, f\}$	Y	{a, c, f}	1

10 Hidden Markov Models (8 points)

Consider a hidden Markov model illustrated as the figure shown below, which shows the hidden state transitions and the associated probabilities along with the initial state distribution. We assume that the state dependent outputs (coin flips) are governed by the following distributions

$P(x = heads|s = 1) = 0.51$
$P(x = heads|s = 2) = 0.49$
$P(x = tails|s = 1) = 0.49$
$P(x = tails|s = 2) = 0.51$

In other words, our coin is slightly biased towards *heads* in state 1 whereas in state 2 *tails* is a somewhat more probable outcome.



(a) Now, suppose we observe three coin flips all resulting in *heads*. The sequence of observations is therefore *heads*; *heads*; *heads*. What is the most likely state sequence given these three observations? (It is not necessary to use the Viterbi algorithm to deduce this, nor any subsequent questions).

2,2,2

The probabilities of outputting head are nearly identical in two states and it is very likely that the system starts from state 2 and stay there. It loses a factor of 9 in probability if it ever switchs to state 1.

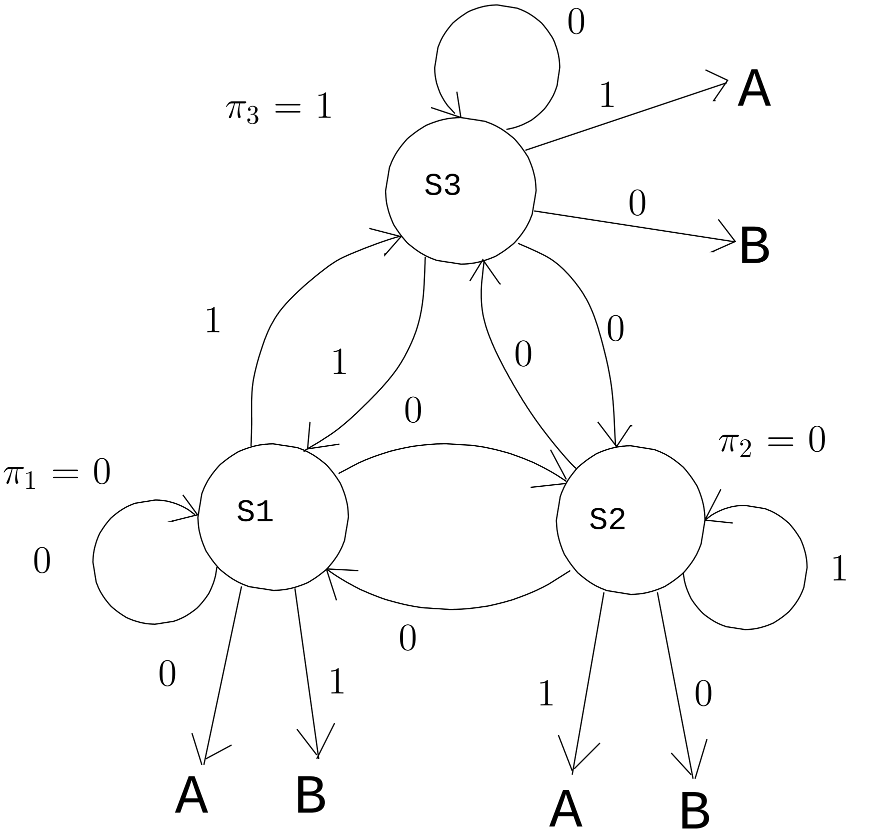
(b) What happens to the most likely state sequence if we observe a long sequence of all heads (e.g., 10^6 heads in a row)?

2,1,1,1,. . .

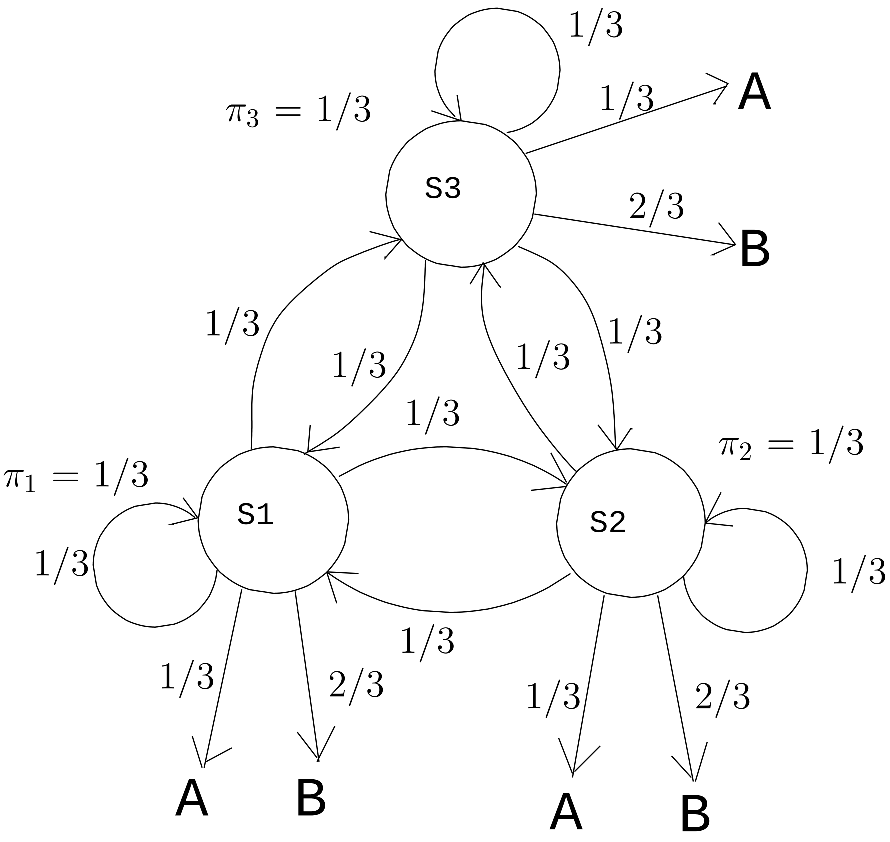
When the number of continuous observations of heads increases, the pressure for the system to switch to state 1 also increases, as state 1 has a slight advantage per observation. Eventually the switch will take place and then there's no benefit from ever switching back to state 2. The cost of the transition switching from state 2 to state 1 is the same regardless of when it takes place. But switching earlier is better than later, since the likelihood of observing the long sequence of all heads is greater. However, it is somewhat better to go via state 2 initially and switch right after (0.99*0.49*0.1 ...) rather than start from state 1 to begin with (0.01*0.51*0.9 ...).

(c) Consider the following 3-state HMM, π_1, π_2 and π_3 are the probabilities of starting from each state $S1$, $S2$ and $S3$. Give a set of values so that the resulting HMM maximizes the likelihood of the output sequence ABA.

There are many possible solutions, and they are all correct as long as they output ABA with probability 1, and the parameter settings of the models are sound. Here is one possible solution:



(d) We're going to use EM to learn the parameters for the following HMM. Before the first iteration of EM we have initialized the parameters as shown in the following figure. **(True or [✓ False])** For these initial values, EM will successfully converge to the model that maximizes the likelihood of the training sequence ABA.



Note the symmetry of the initial set of values over S_1, S_2 and S_3. After each EM iteration, the transition matrix will keep the same ($a_{ij} = 1/3$). The observation matrix may change, but the symmetry still holds ($b_i(A) = b_j(A)$).

(e) **([✓ True] or False)** In general when are trying to learn an HMM with a small number of states from a large number of observations, we can almost always increase the training data likelihood by permitting more hidden states.

To model any finite length sequence, we can increase the number of hidden states in an HMM to be the number of observations in the sequence and therefore (with appropriate parameter choices) generate the observed sequence with probability 1. Given a fixed number of finite sequences (say n), we would still be able to assign probability $1/n$ for generating each sequence. This is not useful, of course, but highlights the fact that the complexity of HMMs is not limited.