Massachusetts Institute of Technology
Department of Electrical Engineering and Computer Science

6.036 Introduction to Machine Learning

Final exam (May 18, 2016)

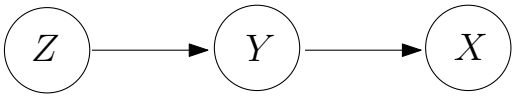
Your name & ID: _____

- This is a closed book exam
- You do not need nor are permitted to use calculators
- The value of each question – number of points awarded for full credit – is shown in parenthesis
- The problems are not necessarily in any order of difficulty. We recommend that you read through all the problems first, then do the problems in whatever order suits you best.
- Record all your answers in the places provided

Prob 1	Prob 2	Prob 3	Prob 4	Prob 5	Prob 6	Prob 7	Total
8	16	19	13	12	12	5	85

Problem 1 This problem revisits some basic questions about Bayesian Networks.

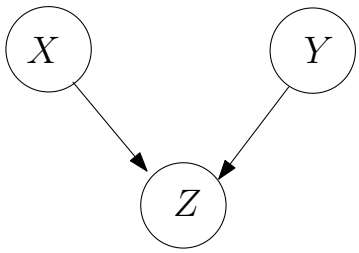
(1.1) **(4 points)** Consider the Bayes Net on three variables X, Y, Z defined by the figure below.



(a) Is $P(X|Y,Z) = P(X|Y)$? (Y/N) ( )

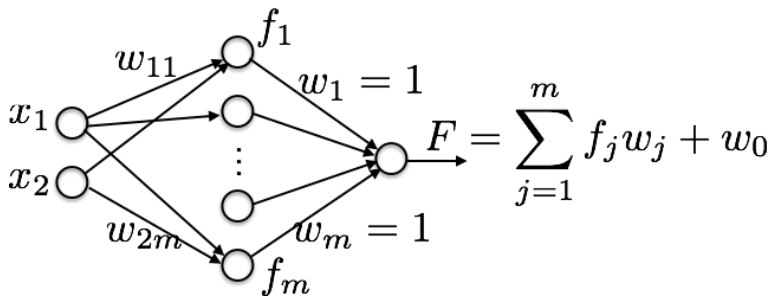
(b) Can we conclude $P(Z|Y,X) = P(Z|Y)$? Briefly justify your answer.

(1.2) **(4 points)** Suppose we have the same three random variables but now they are related in a different way.



For example, the value of X could be a yes/no suggestion from an advisor, and Y value (also yes/no) from another advisor. The president Z decides yes if both advisors agree (say yes), otherwise no. In this specific case, are X and Y independent given $Z = no$? Briefly justify your answer.

Problem 2 Consider a simple two layer neural network for classifying points on the plane. Our network has additional constraints beyond the two-layer architecture. The main constraint is that all the incoming weights to the output layer, w_j, $j = 1, \ldots, m$, are set equal to one, save for the offset parameter w_0 which remains adjustable. The hidden layer units can be chosen arbitrarily, including their number m.



The weights w_{ij}, $i = 0, 1, 2$, $j = 1, \ldots, m$, can be chosen as needed where w_{0j} is the offset parameter for the j^{th} hidden unit.

(2.1) **(2 points)** If the activation function is sign$(\cdot)$, hidden units act as linear classifiers and can be drawn graphically as such. Write down the normal vector to the decision boundary of the linear classifier corresponding to the i^{th} hidden unit?

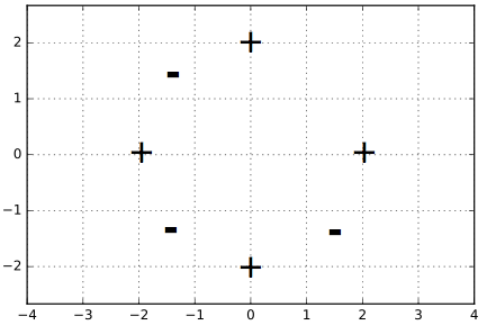
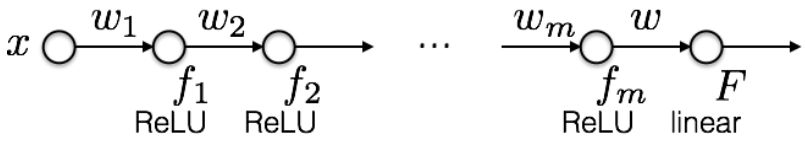


Figure NN1: training set of points

(2.2) **(6 points)** Figure NN1 shows the points we wish to classify correctly. Please draw graphically (as linear classifiers, including orientation) the smallest number of hidden units that enable our constrained output layer to classify the points correctly. For this part you must assume that hidden units are ReLU units.

(2.3) **(4 points)** Neural networks are powerful but can be challenging to train. Consider a simple deep architecture shown below where there is a single unit in each layer.



Each unit uses ReLU activation such that $f_i = \text{ReLU}(w_i f_{i-1} + w_{i0})$, $i = 1, \ldots, m$, where $f_0 = x$. The output unit is linear $F = w f_m + w_0$. For a given input x, we observe target output y, and measure loss $\text{Loss}(F, y)$, where F is the activation of the final linear unit in response to x. In order to train these models with gradient descent, we must be able to calculate gradients. Let

$$d_i = \frac{\partial}{\partial f_i} \text{Loss}(F, y), \quad i = 1, \ldots, m \tag{1}$$

Write down a gradient descent update rule for parameter w_1 using d_i, $i = 1, \ldots, m$.

(2.4) **(4 points)** Suppose $\frac{\partial}{\partial F} \text{Loss}(F, y) = 1$, i.e., we didn't quite predict the response correctly. In this case, which of the following statements are necessarily true in our deep architecture? Check all that apply.

( ) $d_{i-1} = w_i \, f_i \, d_i, \quad i = 2, \ldots, m$
( ) $d_{i-1} = w_i \, [\![\, f_i \geq 0 \,]\!] \, d_i, \quad i = 2, \ldots, m$
( ) $d_m = w$
( ) $d_1 \to 0$ as m increases (vanishing gradient)

(2)

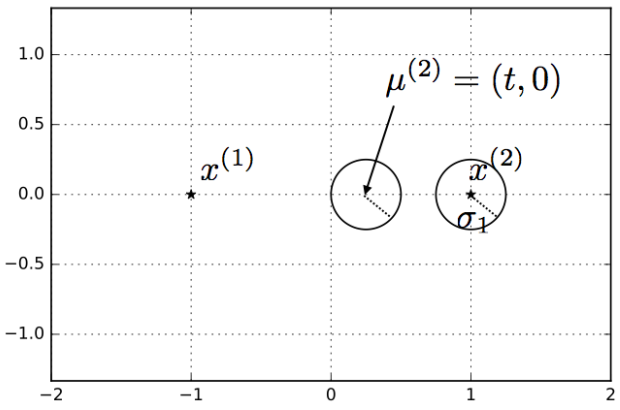
Problem 3

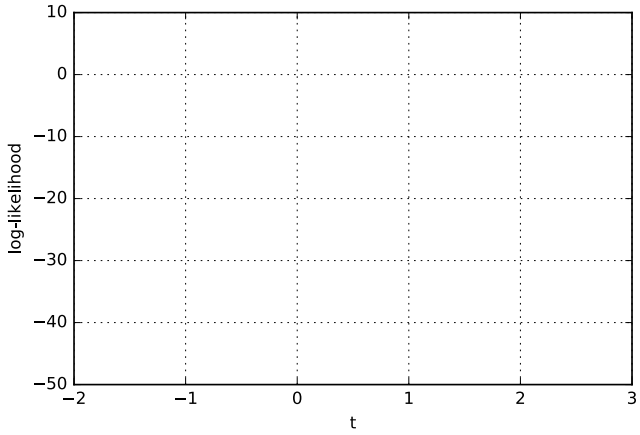
(3.1) **(3 points)** Suppose we fit the mean and the variance of a *single* two-dimensional spherical Gaussian distribution based on points $x^{(1)} = (-1, 0)$ and $x^{(2)} = (1, 0)$. What is the maximum likelihood estimate of the variance of this Gaussian?

(3.2) **(6 points)** Let's consider a two component mixture of spherical Gaussians, i.e.,

$$P(x; \theta) = \pi_1 N(x; \mu^{(1)}, \sigma_1^2 I) + \pi_2 N(x; \mu^{(2)}, \sigma_2^2 I) \tag{3}$$

where we set $\pi_1 = \pi_2 = 0.5$, $\sigma_1 = \sigma_2 = 1/4$, and $\mu^{(1)} = (1, 0)$. We only vary $\mu^{(2)}$ by moving it along the horizontal axis, i.e., we set $\mu^{(2)} = (t, 0)$ where t varies. Figure below (left) illustrates the setting.





The log-likelihood of the data, i.e., points $x^{(1)} = (-1, 0)$ and $x^{(2)} = (1, 0)$, under a single Gaussian $N(x; \mu^{(1)}, \sigma_1^2 I)$ is approximately -30. Qualitatively sketch the log-likelihood of the data under the two component mixture model as a function of t where $\mu^{(2)} = (t, 0)$. Please use the figure above (right).

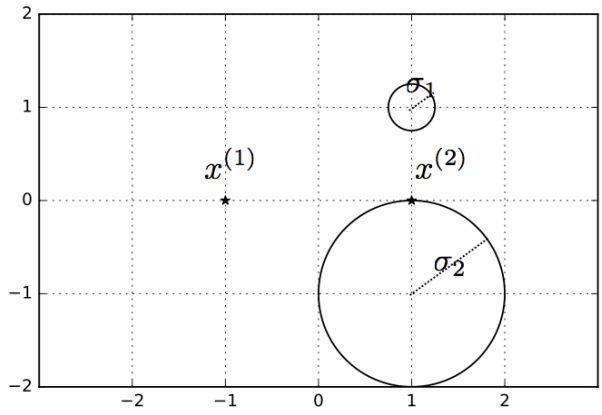


Figure MIX: Initialization of the two component mixture model

(3.3) **(4 points)** We now move on to estimating a two component mixture with a different initialization. The initialization is shown in Figure MIX. Specifically, $\mu^{(1)} = (1, 1)$, $\mu^{(2)} = (1, -1)$, $\sigma_1 = 1/4$, $\sigma_2 = 1$, and $\pi_1 = \pi_2 = 0.5$. Define

$$p(j|x^{(i)}) = \frac{\pi_j N(x^{(i)}; \mu^{(j)}, \sigma_j^2 I)}{\sum_{l=1,2} \pi_l N(x^{(i)}; \mu^{(l)}, \sigma_l^2 I)} \tag{4}$$

Which of the following statements are true for first E-step of the EM algorithm? Check all that apply.

$$\begin{array}{ll} (\quad) & p(1|x^{(1)}) > p(2|x^{(1)}) \\ (\quad) & p(1|x^{(2)}) > p(2|x^{(2)}) \\ (\quad) & p(2|x^{(1)}) > p(2|x^{(2)}) \\ (\quad) & \sum_{i=1,2} p(1|x^{(i)}) < 1 \end{array} \tag{5}$$

(3.4) **(6 points)** Consider the same mixture as in the previous question. Let $\hat{\mu}^{(1)}$, $\hat{\mu}^{(2)}$, $\hat{\sigma}_1$, $\hat{\sigma}_2$, and $\hat{\pi}_1$, $\hat{\pi}_2$ be the parameters after a single M-step. Which of the following statements are true? Check all that apply.

$$\begin{array}{ll} (\quad) & \hat{\mu}_1^{(2)} < 0 \text{ (horizontal component)} \\ (\quad) & \hat{\mu}^{(1)} \approx (1, 0) \\ (\quad) & \hat{\pi}_1 \approx 0 \\ (\quad) & \sigma_1 > \hat{\sigma}_1 \text{ (before vs after the 1st update)} \\ (\quad) & \sigma_2 > \hat{\sigma}_2 \text{ (before vs after the 1st update)} \\ (\quad) & \hat{\mu}^{(2)} = (0, 0) \text{ when EM converges} \end{array} \tag{6}$$

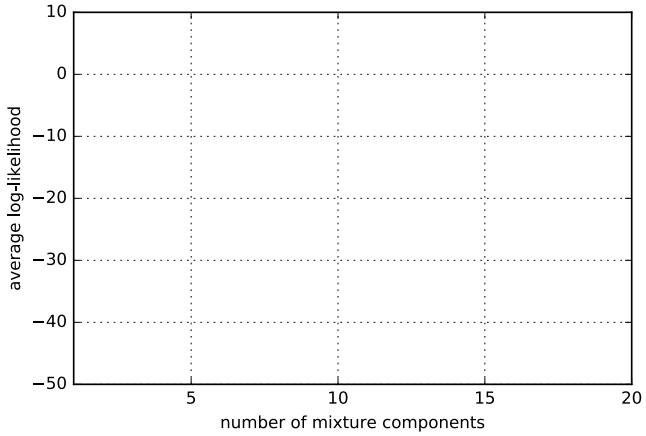
Problem 4 Understanding generalization is one of the core problems in machine learning. Here we scratch the surface a bit in the context of Gaussian mixture models. Suppose we have a training set with n examples $S_n = \{x^{(i)}, i = 1, \ldots, n\}$ sampled from some underlying distribution we don't know. Luckily, we can assume that the training and test examples are sampled from the same underlying distribution. Our goal is to find a mixture model that generalizes well to test examples.

We will measure the performance in terms of the average log-likelihood of data, i.e.,

$$l_n(\hat{\theta}_k) = \frac{1}{n} \sum_{i=1}^{n} \log P(x^{(i)}; \hat{\theta}_k) \tag{7}$$

where $P(x; \hat{\theta}_k)$ refers to a k-component mixture model where the associated parameters $\hat{\theta}_k$ were estimated from the training set via the EM-algorithm. The average test log-likelihood is measured in the same way, only with respect to test examples that were not available during training. We denote this average test log-likelihood as $l(\hat{\theta}_k)$.

(4.1) **(4 points)** In the plot below, qualitatively sketch how training and test log-likelihoods, $l_n(\hat{\theta}_k)$ and $l(\hat{\theta}_k)$, behave as a function of the number of mixture components or k.



(4.2) **(2 points)** Is it possible that the two curves would cross for some choice of training set? (Y/N) **( )**

(4.3) **(3 points)** Which of the following statements are true as n (the size of the training set) increases? Select all that apply.

( ) $l_n(\hat{\theta}_k)$ would typically increase, for any k
( ) $l(\hat{\theta}_k)$ would typically increase, for any k (8)
( ) $|l(\hat{\theta}_k) - l_n(\hat{\theta}_k)|$ would typically decrease, for any k

(4.4) **(4 points)** Our goal here is to estimate a mixture model that generalizes well. We make use of three sub-routines: 1) Init(k, data) that returns a randomly initialized Gaussian mixture model with k components with the help of 'data'; 2) EM$(\text{mix}, \text{data})$ which returns a trained mixture model based on 'data' and the initialization 'mix'; and 3) Eval$(\text{mix}, \text{data})$ returns the average log-likelihood of 'data' for a specific mixture model 'mix'. The pseudo-code below should return two things

(a) the mixture model that is likely to generalize the best

(b) a fair estimate of the resulting average test log-likelihood

The problem is that you only have a training set, randomly divided into five equal size pieces, train$[i]$, $i = 1, \ldots, 5$. You are free to combine pieces into larger sets, e.g., train$[1, 2, 3]$. Fill in the datasets for the pseudo-code as well as the values to return corresponding to part (a) and (b)

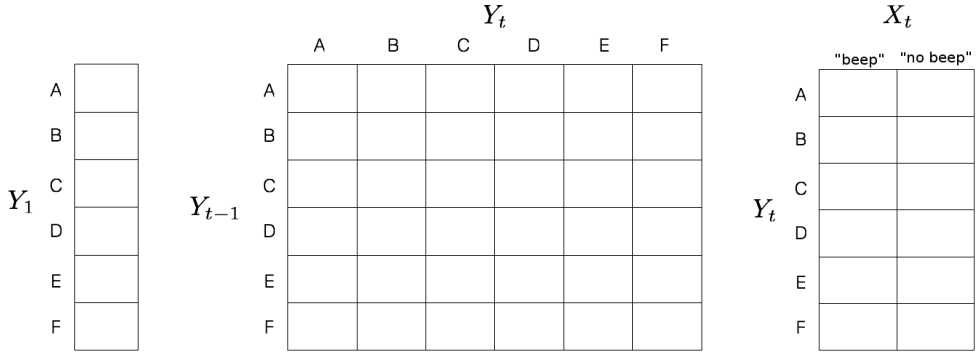
```
For k = 1,...,K

   mix[k] = Init(k, ________________ )  % mixture with k components

   mix[k] = EM(mix[k], ________________ )

   LL[k]  = Eval(mix[k], ________________ )

mix* = _________________________________ % answer to part (a)

LL*  = _________________________________ % answer to part (b)
```

Problem 5 Suppose you are playing an outdoors treasure hunt game. The playing field is divided into a grid as shown below. We know that there is gold buried underneath two of the squares, C and D. You have a heavy metal detector that beeps when on top of a square that may contain gold. However, the metal detector can be misled by the presence of other heavy metals (that we are uninterested in) and can thus beep falsely. From a prior calibration we know that the metal detector is able to detect the gold 75% of the time. The detector never beeps in the absence of gold.

A	B	C
D	E	F

The game becomes interesting because you do not directly observe where you are. With equal probability you start on either square 'A' or 'F'. At each step, you either stay put or move horizontally or vertically (not diagonally), all with equal probability. We will model your location as a hidden state $Y_t \in \{A, B, C, D, E, F\}$, whereas the observation X_t is a "beep" or "no beep".

(5.1) **(4 points)** Specify the initial state, state transition, and the emission probabilities for this HMM.



(5.2) **(4 points)** Suppose you observed the following sequence of sounds from the metal detector: (X_1, X_2, X_3) =("no beep", "beep", "beep"). What are the possible sequences (Y_1, Y_2, Y_3) of locations that you may be in, given the three observations?

(5.3) **(4 points)** Which sequence of locations (Y_1, Y_2, Y_3) is most likely to occur together with (X_1 = "no beep", X_2 = "beep, X_3 = "beep"), and what is the corresponding joint probability $P(Y_1, Y_2, Y_3, X_1, X_2, X_3)$? If the answer isn't unique, select one.

Problem 6 You are running a 3 mile race. Every 10 minutes you must decide whether to walk or run for the next 10 minutes based on your current distance from the start (represented as states 0, 1, and 2 but no actions will be taken from state 3 because you will have already finished). If you walk, you will advance 1 mile over the next 10 minutes. If you run, you have a 50% chance to advance 1 mile and a 50% chance to advance 2 miles over the next 10 minutes. You want to finish the race, but running is tiring and takes effort. You will receive a reward of 10 for finishing the race (ending up in state 3). However, every time you run, you get an additional "reward" -1. You decide to use a Markov Decision Process with γ=0.5 to determine what action you should take from each state. The full table of transition probabilities and rewards is shown below.

s	a	s'	T(s,a,s')	R(s,a,s')
0	WALK	1	1.0	0
1	WALK	2	1.0	0
2	WALK	3	1.0	10
0	RUN	1	0.5	-1
0	RUN	2	0.5	-1
1	RUN	2	0.5	-1
1	RUN	3	0.5	+9
2	RUN	3	1.0	+9

(6.1) **(3 points)** Suppose we initialize $Q_0(s, a) = 0$ for all s ϵ {0, 1, 2} and all a $\in$ {WALK, RUN}. We assume that the values in state $s = 3$ are always zero for any action. Evaluate the Q-values $Q_1(s, a)$ after exactly one Q-value iteration.

a	s=0	s=1	s=2
WALK			
RUN			

(6.2) **(3 points)** What is the ideal policy derived from $Q_1(s, a)$?

$$\begin{aligned} \pi_1^*(s=0) &= \\ \pi_1^*(s=1) &= \\ \pi_1^*(s=2) &= \end{aligned} \tag{9}$$

(6.3) **(3 points)** What are the values $V_1(s)$ using the values of $Q_1(s, a)$ calculated above?

s=0	s=1	s=2

(6.4) **(3 points)** Consider now iterating Q-values one more time to obtain $Q_2(s, a)$. We are only interested in here what happens at $s = 0$. For what range of values of the discount factor $0 \leq \gamma \leq 1$ would the action derived from $Q_2(0, a)$ suggest that we RUN?

Problem 7

(7.1) **(5 points)** Choose T/F for each of the following statements.

( ) While neural networks are powerful, classifiers based on SIFT features still perform better at object recognition tasks

( ) The tasks of tracking people and objects are solved quite well for short time spans but remain challenging for longer trajectories

( ) Bayes filtering is unsuitable for robot localization.

( ) Aerial robots typically make use of laser range finders (rather than cameras) to carve out open space

( ) SLAM stands for simultaneous localization and mapping and is the process by which a robot explores to create a map of its environment

6.036 Official Cheat Sheet

- A (hyper-)plane is a set of points $x \in \mathcal{R}^d$ such that $\theta \cdot x + \theta_0 = 0$. Vector θ is normal to the plane. The signed distance of any point x from the plane is $(\theta \cdot x + \theta_0)/\|\theta\|$. The value of distance is positive on the side where θ points to, and negative on the other side.
- A linear classifier with offset: $h(x;\theta) = \text{sign}(\theta \cdot x + \theta_0)$
- Training error (classification error): $\epsilon_n(h) = \frac{1}{n}\sum_{i=1}^{n}[[y^{(i)} \neq h(x^{(i)})]]$
  The same formula can be used to calculate the test error.
- Loss functions:
  - $z = y(\theta \cdot x + \theta_0)$ (agreement)
  - $\text{Loss}_{0,1}(z) = [[z \leq 0]]$
  - $\text{Loss}_{hinge}(z) = \max\{1 - z, 0\}$
- Passive-aggressive algorithm (no offset): At step k, in response to (x, y), find $\theta^{(k+1)}$ that minimizes $\lambda\|\theta - \theta^{(k)}\|^2/2 + Loss_{hinge}(y\theta \cdot x)$
- SVM: Find a maximum-margin classifier by minimizing

  $\frac{1}{n}\sum_i \text{Loss}_{hinge}(y^{(i)}(\theta \cdot x^{(i)} + \theta_0)) + \frac{\lambda}{2}||\theta||^2$

  which can be done using stochastic gradient descent (Pegasos).
- Linear regression: predict value $\theta \cdot x + \theta_0$ by minimizing $\frac{\lambda}{2}||\theta||^2 + \frac{1}{n}\sum_{i=1}^{n}(y^{(i)} - \theta \cdot x^{(i)} - \theta_0)^2/2$
- Low-rank matrix factorization for collaborative filtering: Minimize

  $J(U, V) = \sum_{(a,i)\in D}(Y_{ai} - [UV^T]_{ai})^2/2 + \frac{\lambda}{2}\sum_{a=1}^{n}\sum_{j=1}^{k}U_{aj}^2 + \frac{\lambda}{2}\sum_{i=1}^{m}\sum_{j=1}^{k}V_{ij}^2$

  Can be solved iteratively by fixing one matrix and using linear regression to find the other.
- Kernels: $K(x, x') = \phi(x) \cdot \phi(x')$

Kernel	form
Linear	$x \cdot x'$
Quadratic	$x \cdot x' + (x \cdot x')^2$
Radial basis	$\exp(-\|\|x - x'\|\|^2/2)$

- Kernel Perceptron (with offset): Cycles through each point t=1,..n and checks if $y^{(t)}(\sum_{i=1}^{n}\alpha_i y^{(i)}[K(x^{(i)}, x^{(t)}) + 1]) \leq 0$. If true, $\alpha_t = \alpha_t + 1$.
- Neural Nets:
  - output of neuron with activation function f is $Z = \sum_{j=1}^{m} f(z_j)V_j + V_0$
  - a unit in layer l with net input $z_j^l = \sum_i f_i^{l-1}w_{ij}^l + w_{0j}^l$ will output $f_j^l = f(z_j^l)$
  - common activation functions include ReLU ($f(z) = \max\{0, z\}$), tanh, and the identity function
  - backpropagation (with respect to inputs z): $\delta_i^{l-1} = f'(z_i^{l-1})\sum_j w_{ij}^l\delta_j^l$
- Generalization:

  In the realizable case, $\epsilon(\hat{h}) \leq \epsilon_n(h) + \frac{\log|H| + \log(\frac{1}{\delta})}{n}$ where $\epsilon_n(h)$ is the training error and H is a finite set of classifiers.

  In the non-realizable case, we obtain a weaker bound: $\epsilon(\hat{h}) \leq \epsilon_n(h) + \sqrt{\frac{\log|H| + \log(\frac{1}{\sigma})}{2n}}$.

  When H is not finite, $\log|H|$ will be roughly speaking replaced by the growth function $\log N_H(n)$ which relates to the VC-dimension.

  $BIC(D;\theta) = l(D;\theta) - \frac{\text{number of params}}{2}\log(n)$

where D is the data containing n examples.

- HMM
  - $P(X_1, ..., X_T) = P(X_1)\Pi_{t=2}^{T} P(X_t|X_{t-1})$
  - $P(Y_{1:T}, X_{1:T}) =$ $P(Y_1)P(X_1|Y_1|)\Pi_{t=2}^{T} P(Y_t|Y_{t-1})P(X_t|Y_t)$
- Q-Value Iteration Algorithm:
  1. $Q_0(s, a) = 0$
  2. $Q_{i+1}(s, a) =$ $\sum_{s'} T(s, a, s')[R(s, a, s') + \gamma \max_{a'} Q_i(s', a')]$
- Value Iteration Algorithm:
  1. $V_0(s) = 0$
  2. $V_{i+1}(s) =$ $\max_a \left[\sum_{s'} T(s, a, s')[R(s, a, s') + \gamma V_i(s')\right]$

  Note that $V_i(s') = \max_{a'} Q_i(s', a')$
- Q-learning

  Model-free estimation (to avoid explicitly computing T,R) $Q(s, a) \leftarrow$ $Q(s, a) + \alpha[R(s, a, s') + \gamma \max_{a'} Q(s', a') - Q(s, a)]$
- K-Means

  $\text{cost}(\mu^{(1)}, \ldots, \mu^{(k)}) =$ $\sum_{i=1}^{n} \min_{j=1,...,k} \|x^{(i)} - \mu^{(j)}\|^2$
  1. initialize $\mu^{(1)}, ..., \mu^{(k)}$
  2. $\delta(j|i) = [[\, j = \text{argmin}_l \|x^{(i)} - \mu^{(l)}\|^2 \,]]$
  3. $\hat{\mu}^{(j)} = \frac{1}{\sum_{i=1}^{n} \delta(j|i)} \sum_{i=1}^{n} \delta(j|i) x^{(i)}$
- EM for Gaussians:
  1. initialize $\theta = \{p_1, ..., p_k, \mu^{(1)}, ..., \mu^{(k)}, \sigma_1^2, ..., \sigma_k^2\}$
  2. E-Step: $p(j|i) = \frac{p_j N(x^{(i)}; \mu^{(j)}, \sigma_j^2 I)}{\sum_{z=1}^{k} p_z N(x^{(i)}; \mu^{(z)}, \sigma_z^2 I)}$
  3. M-step: $\max_\theta \sum_{i=1}^{n} \sum_{j=1}^{k} p(j|i) \log[p_j N(x^{(i)}; \mu^{(j)}, \sigma_j^2 I)]$, giving
     - $p_j = \frac{\sum_{i=1}^{n} p(j|i)}{n}$
     - $\hat{\mu}^{(j)} = \frac{1}{\sum_{i=1}^{n} p(j|i)} \sum_{i=1}^{n} p(j|i) x^{(i)}$
     - $\hat{\sigma}_j^2 =$ $\frac{1}{d \sum_{i=1}^{n} p(j|i)} \sum_{i=1}^{n} p(j|i) \|x^{(i)} - \hat{\mu}^{(j)}\|^2$
- max-likelihood estimates for $N(x; \mu, \sigma^2 I) = \frac{1}{(2\pi\sigma^2)^{d/2}} \exp(-\frac{1}{2\sigma^2}\|x - \mu\|^2)$
  - If $x \in R$ (1-dimensional): $\hat{\mu} = \frac{1}{n} \sum_{i=1}^{n} x^{(i)}$, $\hat{\sigma}^2 = \frac{1}{n} \sum_{i=1}^{n} (x^{(i)} - \hat{\mu})^2$
  - If $x \in R^d$ (d-dimensional): $\hat{\mu} = \frac{1}{n} \sum_{i=1}^{n} x^{(i)}$, $\hat{\sigma}^2 = \frac{1}{dn} \sum_{i=1}^{n} \|x^{(i)} - \hat{\mu}\|^2$
- log-likelihood $\ell(S_n; \theta) = \sum_{i=1}^{n} \log P(x^{(i)}; \theta)$