# CS 344 (Spring 2018): End-semester Examination

Instructor: Shivaram Kalyanakrishnan

9.30 a.m. – 12.30 p.m., May 3, 2018, LA 301

Total marks: 35

**Note.** Provide brief justifications and/or calculations along with each answer to illustrate how you arrived at the answer.

**Question 1.** This question (which ought to be easy!) tests your AI vocabulary and high-level retention of topics discussed in this course. Each of the 12 entries shown in the table below is an important idea, concept, or algorithm.

| **Concept** | **Topic** (Pick from: Clustering, Decision trees, Search, Bayes Nets, Markov Decision Problems, Neural networks) |
|---|---|
| Gibbs sampling | |
| Activation function | |
| Value Iteration | |
| Discount factor | |
| Iterative deepening | |
| Conditional independence | |
| Alpha-Beta pruning | |
| D-separation | |
| Q-learning | |
| Admissible heuristic | |
| A$^\star$ | |
| Backpropagation | |

Associate each entry with the topic to which it is *most* relevant. Pick topics from among the six choices shown in the "Topic" column. Not all the topics provided need find matches. [6 marks]

**Question 2.** Consider an optimal 2-clustering of a data set $x^1, x^2, \ldots, x^n$, $n \geq 2$, where for $i \in \{1, 2, \ldots, n\}, x^i \in \mathbb{R}^d$, where $d \geq 1$. Without loss of generality, assume that for some $m \in \{1, 2, 3, \ldots, n-1\}$, the points $x^1, x^2, \ldots, x^m$ are assigned to the first cluster, and the points $x^{m+1}, x^{m+2}, \ldots, x^n$ are assigned to the second cluster. Assume that the $n$ points are all distinct, and no point is equidistant to both cluster centres.

Show that there exists a hyperplane of the form $a \cdot x + b = 0$, where $a \in \mathbb{R}^d, b \in \mathbb{R}$, such that all the points in the first cluster $(x^1, x^2, \ldots, x^m)$ lie to one side of the hyperplane, and all the points in the second cluster $(x^{m+1}, x^{m+2}, \ldots, x^n)$ lie to the other side of the hyperplane. You should express $a$ and $b$ in terms of $x^1, x^2, \ldots, x^n$, and $m$. Notice that multiplying $a$ and $b$ by the same scalar leaves the hyperplane unchanged—hence there are multiple correct answers. [6 marks]

**Question 3.** We consider a 2-class supervised learning problem, for which the training data set is $D$. A certain randomised learning algorithm $L$ has the property that on a given run, the model $M$ it produces makes a correct prediction with probability $p$ independently for each point in $D$, and an incorrect prediction with probability $q = 1 - p$. $L$ is a *weak learner*, in the sense that it performs better than random guessing. In other words, its expected accuracy in predicting the label of a point picked uniformly at random from $D$, which is $p$, exceeds $\frac{1}{2}$ (equivalently, $q < \frac{1}{2}$).

A meta-learning algorithm $L(N)$, where $N \geq 1$, uses $L$ as a subroutine to independently train $N$ separate models $M_1$, $M_2$, ..., $M_N$. Since $L$ is randomised, these models could all be different. However, note that a run of $L$ satisfies the constraint stated above—of classifying each point from $D$ correctly with probability $p$. The output computed by $L(N)$ is the majority of the outputs given by $M_1, M_2, \ldots, M_N$. To avoid tie-breaking, let us assume $N$ is odd, and take $N = 2n + 1$, where $n \geq 0$.

3a. What is the expected error of $L(2n + 1)$ in predicting the label of a point picked uniformly at random from $D$? [3 marks]

3b. Show that the error obtained in 3a can be made arbitrarily small by making $n$ sufficiently large. One way to do so would be to establish an upper bound on the error (also as a function of $n$) whose limit (as $n \to \infty$) is 0. Remember that $p > \frac{1}{2}$ and $q < \frac{1}{2}$. [3 marks]

**Question 4.** Your task in this question is to express the Boolean function $y = x_1 \oplus x_2 \oplus x_3$, where $x_1, x_2, x_3, y \in \{0, 1\}$, using a neural network. This is the *parity* function: the output $y$ is 1 if exactly one or three of $x_1$, $x_2$, and $x_3$ are 1; $y$ is 0 otherwise. The truth table of the function is shown below.

| $x_1$ | $x_2$ | $x_3$ | $y$ |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |

Your neural network to implement this function must take three inputs: $x_1$, $x_2$, and $x_3$, and produce a single output $y$. Each node in the network must implement the *sign* function; recall that

$$sign(\alpha) = \begin{cases} 1, & \text{if } \alpha \geq 0; \\ 0, & \text{otherwise.} \end{cases}$$

Nodes can also take in a bias as an input.

To obtain full marks, your neural network must achieve the desired input-output response, and be limited to a *single* hidden layer: that is, a network of the form 3-$X$-1 (with $X$ nodes in the hidden layer). If you use more than one hidden layer, you will receive 4 marks if the network achieves the correct input-output response. [6 marks]

**Question 5.** "$k$ Nearest Neighbours" ($k$-NN) is a conceptually simple algorithm for supervised learning. Assume we are solving a 2-class classification problem in $\mathbb{R}^d$ for $d \geq 1$. We are given a training data set $D = \{(x^1, y^1), (x^2, y^2), \ldots, (x^n, y^n)\}$, where $n \geq 2$; for $i \in \{1, 2, \ldots, n\}, x^i \in \mathbb{R}^d$ and $y^i \in \{0, 1\}$. Assume the data set has at least one example from each class (otherwise the problem would be trivial).

There is no actual "learning" involved in $k$-NN: the learned "model" is merely the data set $D$ itself. When a label has to be computed for a new (test) point $x \in \mathbb{R}^d$, the $k$ entries of $D$ closest to $x$, say in terms of Euclidean distance, are looked up, and their majority label is returned as the answer. As an example, consider $k = 3$, with $x^5$, $x^{23}$, and $x^{45}$ being the 3 closest entries to the query point $x$ in $D$ . The label predicted for $x$ is 0 if two or more among $y^5$, $y^{23}$, and $y^{45}$ are 0; otherwise the label predicted is 1. Any ties that might arise while picking the $k$ nearest neighbours and their majority prediction are broken arbitrarily.

5a. How do you expect the accuracy of the $k$-NN method to vary with $k$, assuming the data is "naturally occurring" (think of the MNIST data set as an example)? What would the accuracy on the training data (that is, $D$) be for small, intermediate, and large values of $k$? What would the accuracy on test data (picked independently of $D$) be for small, intermediate, and large values of $k$? [2 marks]

5b. Describe an $O(nk)$-time procedure to compute the prediction for a given test query $x$? [1 mark]

5c. Consider the special case of $d = 1$ (that is, 1-dimensional data). Can you compute the prediction more efficiently? What is the best time complexity (in terms of $n$ and $k$) that you can achieve? Describe the corresponding procedure. [3 marks]

You do not have to write explicit pseudocode for 5b and 5c, provided your clearly describe the sequence of steps to be executed.

**Question 6.** The following questions test your knowledge of aspects related to AI in the real world. Keep your answers crisp. 4–6 lines will suffice for 6a and 6b, and 2–3 lines for 6c.

6a. What is the principle behind Tim Berners-Lee's star-rating system for open data? Briefly describe the rating system as a part of your answer. [2 marks]

6b. Which area of machine learning constituted the technical core in training the AlphaGo program to play the game of Go? AI achieved human-level competence at Chess in the late 1990's. Why do you think it took nearly two more decades for AI to achieve human-level competence at Go? [2 marks]

6c. Why was the UK-based company *Cambridge Analytica* in the news some months back? What relevance does the related episode bear to modern AI? [1 mark]

# Solutions

**1.**

| **Concept** | **Topic** (Pick from: Clustering, Decision trees, Search, Bayes Nets, Markov Decision Problems, Neural networks) |
|---|---|
| Gibbs sampling | Bayes Nets |
| Activation function | Neural networks |
| Value Iteration | Markov Decision Problems |
| Discount factor | Markov Decision Problems |
| Iterative deepening | Search |
| Conditional independence | Bayes Nets |
| Alpha-Beta pruning | Search |
| D-separation | Bayes Nets |
| Q-learning | Markov Decision Problems |
| Admissible heuristic | Search |
| A$^\star$ | Search |
| Backpropagation | Neural networks |

**2.** Let $\mu^1$ and $\mu^2$ be the cluster centres of the optimal clustering. Since it is an optimal clustering, the centres have to be the centroids of the points belonging to the clusters (otherwise SSE can be further reduced). Thus,

$$\mu^1 = \frac{1}{m}\sum_{i=1}^{m} x^i \text{ and } \mu^2 = \frac{1}{n-m}\sum_{i=m+1}^{n} x^i.$$

Again, since the clustering is optimal, for every point $i \in \{1, 2, \ldots, m\}$ in the first cluster, we have $\|x^i - \mu^1\| < \|x^i - \mu^2\|$ (otherwise SSE can be reduced by assigning $x^i$ to the second cluster). Observe that

$$\begin{aligned}
&\|x^i - \mu^1\| < \|x^i - \mu^2\| \\
&\iff \|x^i - \mu^1\|^2 < \|x^i - \mu^2\|^2 \\
&\iff (x^i - \mu^1)\cdot(x^i - \mu^1) < (x^i - \mu^2)\cdot(x^i - \mu^2) \\
&\iff x^i \cdot x^i + \mu^1 \cdot \mu^1 - 2x^i \cdot \mu^1 < x^i \cdot x^i + \mu^2 \cdot \mu^2 - 2x^i \cdot \mu^2 \\
&\iff (\mu^1 - \mu^2)\cdot\left(\frac{\mu^1 + \mu^2}{2} - x^i\right) < 0.
\end{aligned}$$

By similar reasoning, for $i \in \{m+1, m+2, \ldots, n\}$ in the second cluster, we have $(\mu^1 - \mu^2)\cdot(\frac{\mu^1+\mu^2}{2} - x^i) > 0$. The clusters are seen to be separated by the hyperplane $(\mu^1 - \mu^2)\cdot(\frac{\mu^1+\mu^2}{2} - x^i) = 0$, which, in keeping with intuition, is the perpendicular bisector of the line joining the cluster centres. The hyperplane can equivalently be written down as

$$(\mu^2 - \mu^1)\cdot x + \frac{\mu^1 \cdot \mu^1 - \mu^2 \cdot \mu^2}{2} = 0.$$

**3a.** A point will be misclassified by $L(2n+1)$ if and only if the point is misclassified by $n+1$ or more classifiers among $M_1, M_2, \ldots, M_{2n+1}$. The probability that it will be misclassified by exactly $i$ classifiers, $i \in \{1, 2, \ldots, 2n+1\}$ is $\binom{n}{i} q^i p^{2n+1-i}$, and so the expected error of $L(2n+1)$ is

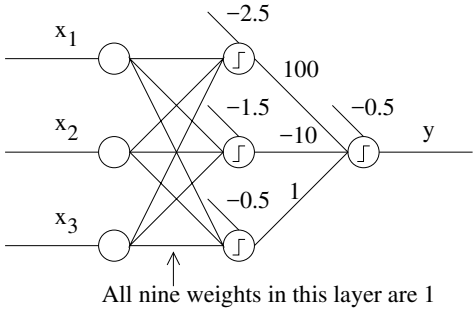
$$\sum_{i=n+1}^{2n+1} \binom{n}{i} q^i p^{2n+1-i}.$$

**3b.** Since $p > q$, we can upper-bound each term by keeping the exponent of $q$ as $n+1$ and that of $p$ as $n$, which makes the error upper-bounded by

$$\sum_{i=n+1}^{2n+1} \binom{n}{i} q^{n+1} p^n = \frac{2^{2n+1}}{2} q^{n+1} p^n = q(4pq)^n.$$

Since $p > \frac{1}{2}$ and $q = 1 - p$, it follows that $4pq < 1$, and so $\lim_{n\to\infty} q(4pq)^n = 0$. Therefore, for any $\epsilon > 0$, there will exist an $n \geq 0$ such that the expected error of $L(2n+1) < \epsilon$.

**4.** The figure below shows one possible way to encode the parity function on 3 inputs as a 3-3-1 neural network. The weights connecting the input layer with the first hidden layer are all 1; the first hidden layer thresholds the sum of the inputs. Observe that when $x_1 + x_2 + x_3 = 0$, no node in the hidden layer fires (that is, has a positive output). When $x_1 + x_2 + x_3 = 1$, only the lowest node in the hidden layer fires. When $x_1 + x_2 + x_3 = 2$, both the lowest and the middle nodes in the hidden layer fire. When $x_1 + x_2 + x_3 = 3$, all the nodes in the hidden layer fire.




Since the output should be 1 only when $x_1 + x_2 + x_3 = 3$ or $x_1 + x_2 + x_3 = 1$, we weight the outputs of the hidden layer such that the "highest" firing node determines the eventual output $y$.

The solution presented above is by no means unique; any encoding using a 1-hidden-layer network that implements the parity function correctly will receive full marks.

**5a.** On training data, $k = 1$ will give the highest accuracy (of 100%), as it will always give exact predictions. As $k$ is increased, the accuracy is likely to decrease. On test data, $k = 1$ (and very low values of $k$) are not likely to give good predictions, as they will tend to overfit the training data. Some intermediate values of $k$ are likely to make the best predictions. Very large values of $k$ imply a very low-complexity model, which might not be capable of making nuanced predictions.

**5b.** The obvious way to compute the label is to find the nearest neighbour of $x$ (in $O(n)$ time), then the second nearest neighbour of $x$ (in $O(n)$ time), ..., then the $k$-th nearest neighbour of $x$ (in $O(n)$ time); and then to find the majority label among these neighbours (in $O(k)$ time). The total complexity is upper-bounded by $O(nk)$, which can be improved by using better data structures to $O(n \log(k) + k)$.[1]

**5c.** If $d = 1$, we can sort the $n$ points as a preprocessing step (taking $O(n \log(n))$ time). Thereafter, finding the $k$ nearest neighbours of $x$ can be done by first isolating the "left" and "right" neighbours of $x$ (by binary search, in $O(\log(n))$ time), and then expanding out to the left and the right until the $k$ nearest neighbours have been identified (in $O(k)$ time). The overall time complexity for processing a query is therefore only $O(\log(n) + k)$, assuming the data is sorted (which can be done in $O(n \log(n))$ time).

The main difference between $d = 1$ and higher values of $d$ is that in the latter case, it is not possible to "sort" the data in a manner that enables efficient querying. In practice, though, one could use sophisticated randomised approaches for finding nearest neighbours in higher dimensions with a probabilistic guarantee.

**6a.** The principle underlying Tim Berners-Lee's star rating system for open data is that not all data sets available for public access are equally easy to use. The rating system associates a "star-rating" with each data set that is consistent with the ease with which its contents can be accessed and purposefully utilised. 1 star is given if the data is publicly accessible with an open licence; 2 stars if it is in some structured format (such as a table); 3 stars if the format is non-proprietary; eventually 5 stars if the data set is explicitly linked with other data tables.

**6b.** AlphaGo was trained mainly through the application of Reinforcement learning. Self-play was used to generate training trajectories. Go has an orders-of-magnitude-larger state space and a larger average branching factor than Chess. Moreover, humans have not been able to understand and articulate concepts underlying Go strategy as well as they have been able to for Chess. For these reasons, it took AI many more years to reach human-level competence at Go, after having done so at Chess.

**6c.** The company got possession of the Facebook data of millions of Facebook users through an unauthorised chain of transactions. The company is thought to have analysed this data and provided consulting services to entities such as political parties. Modern AI is driven by data; the episode highlights the dangers that can result from data theft.

[1] Note that the dependence on $d$ has not been included in the $O()$ notation. Equivalently, we have assumed that the distance between two points in $\mathbb{R}^d$ can be calculated in $O(1)$ time.