Final Exam Review

10-701: Introduction to Machine Learning

4/26/2024

1 Unsupervised Learning

1.1 Clustering

1. For each of the **True or False** questions below, select the correct answer and briefly justify your selection in 1-2 concise sentences.

   (a) For a fixed dataset and k, the k-means algorithm will always produce the same result if the initial centers are the same.

   ○ True

   ○ False

   True: the k-means algorithm is deterministic.

   (b) The k-means algorithm will always converge to the globally optimal solution.

   ○ True

   ○ False

   False: this depends on the initialization. Poor initializations can lead to local minima.

(c) In the k-means algorithm, the objective function's value can increase, decrease or stay the same after each iteration.

○ True

○ False

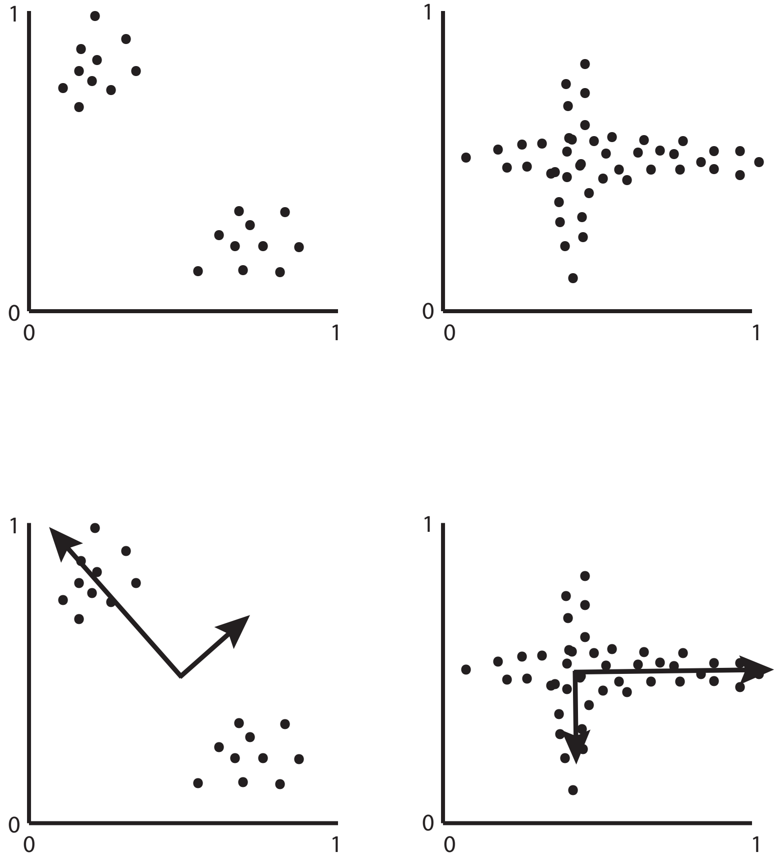
False: the objective function's value can never increase from one iteration to the next.

2. **Short answer:** In the setting of expectation-maximization for Gaussian mixture models, define the marginal log-likelihood (either in words or mathematically) and descrbie why we optimize the marginal log-likelihood instead of the complete log-likelihood of the dataset.

The marginal log-likelihood is the probability of the data points (the $x^{(i)}$'s) given the parameters under the modelling assumptions of a Gaussian mixture model. The reason why we optimize the marginal log-likelihood instead of the complete log-likelihood is that complete log-likelihood requires observing the latent cluster assignments, which are unknown and must be inferred from the data.

1.2 Dimensionality Reduction

1. **Drawing:** Consider the following two datasets. On the figures below, draw arrows from the mean of the data to denote the direction and relative magnitudes of the principal components.


1
0
0
1
1
0
0
1
1
0
0
1
1
0
0
1

2. Given a dataset $\mathcal{D}$ consisting of N data points and D features, suppose you use PCA to project the dataset down to $d < D$ dimensions: let E be the squared reconstruction error of this projection.

   (a) **Select one:** Now suppose that you add an extra data point to $\mathcal{D}$ so that $\mathcal{D}'$ consists of $N+1$ data points and D features. You once again use PCA to project $\mathcal{D}'$ to $d < D$ dimensions: let E' be the squared reconstruction error of this new projection. How do E and E' relate to one another?

   ○ $E < E'$

   ○ $E \leq E'$

   ○ $E = E'$

   ○ $E \geq E'$

   ○ $E > E'$

   B

   (b) **Select one:** Now suppose that you use PCA to project the original dataset $\mathcal{D}$ down to $(d+1) < D$ dimensions instead of d dimensions: let E' be the squared reconstruction error of this new projection. How do E and E' relate to one another?

   ○ $E < E'$

   ○ $E \leq E'$

   ○ $E = E'$

   ○ $E \geq E'$

   ○ $E > E'$

   D

3. **Select all that apply:** Recall from lecture that autoencoders are trained by minimizing the reconstruction error between the inputs $\mathbf{x}$ and the corresponding outputs $\mathbf{x}'$. Given a dataset, $\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^N$, which of the following are rational alternative objective functions for training autoencoders with backpropagation?

   □ $\frac{1}{N}\sum_{i=1}^N \|\mathbf{x}^{(i)'}\|_2^2$

   □ $\frac{1}{N}\sum_{i=1}^N \|\mathbf{x}^{(i)'} - \mathbf{x}^{(i)}\|_1$

   □ $\underset{i}{max}\|\mathbf{x}^{(i)'} - \mathbf{x}^{(i)}\|_2^2$

   □ $\frac{1}{N}\sum_{i=1}^N \|\mathbf{x}^{(i)'} - \frac{1}{N}\sum_{i=1}^N \mathbf{x}^{(i)'}\|_2^2$

   □ None of the above.

C; B is not differentiable and thus, not a viable candidate for optimization via backpropagation. A and D are fundamentally different from the reconstruction error and will drive the autoencoder to learn meaningless representations. Note that C is a computationally expensive objective to optimize but is (potentially) feasible given the size of the dataset.

2 Reinforcement Learning

1. Formulate the task of deciding how long to study for each of our two 10-701 exams this semester as a single reinforcement learning problem. Specifically, briefly describe in words the following components:

   (a) **Short Answer:** The state space $\mathcal{S}$.

   (b) **Short Answer:** The action space $\mathcal{A}$; make sure to specify what actions can be taken in any given state.

   (c) **Short Answer:** The reward function or distribution.

   (d) **Short Answer:** The transition function or distribution.

This is a very open-ended question and we should be generous when grading. For all parts there are multiple valid answers but most should involve a temporal state space and correspondingly, a deterministic transition function. The action space should also be varying study times (note that crucially in this question, we're only asking about how long to study for, not how to study). The big differences should be in how granular they choose to define their state and action spaces, e.g., two states, before exam 1 and after exam 1 but before exam 2 vs. weeks or days before each exam; action spaces could reasonably be discrete (e.g., hours of study) or continuous. Rewards should probably be somehow related to the exam grades, stochastic and unknown a priori. Transition functions are probably the most rigid component of this question: given the temporal nature of the state space, for most valid formulations, the transitions should be deterministic and known, simply going from one time-step to the next.

2. **Select one:** Which algorithm is *most* appropriate for solving the problem you defined above? Briefly justify your answer in 2-3 concise sentences.

   ○ Value iteration

   ○ Policy iteration

   ○ Q-learning

   ○ Deep Q-learning

   Again this could vary based on their answer to the previous part but most reasonable formulations will likely use either Q-learning or deep Q-learning as the reward function should be unknown; the choice between Q-learning and deep Q-learning will depend on if they chose to discretize their action space or not. If they formulated their problem to have a known reward (strange but potentially justifiable), then the choice between value and policy iteration could reasonably boil down to how large their state space is: most formulations will have small state spaces so one should prefer policy iteration as the additional overhead of solving the Bellman equations ($O(|\mathcal{S}|^3)$) will be negligible.

3. **Math:** In class we taught $\pi(s)$ as the action taken in state s under deterministic policy π. Now we consider the case of a stochastic policy π, such that $\pi(a \mid s)$ is the probability of taking action a in state s under stochastic policy π. If the current state is s_t, write the expectation of r_{t+1} in terms of $\pi(a \mid s)$ and $p(s', r \mid s, a)$, the probability of transitioning to state s' with reward r, from state s and action a.

$$E[R_{t+1} \mid s_t] = \sum_{a} \pi(a \mid s_t) \sum_{s',r} r p(s', r \mid s_t, a)$$

4. **Select all that apply:** Which of the following are *necessary* conditions for Q-learning to converge to the optimal Q values?

   □ For every state, every valid action is taken at least once.

   □ The discount factor is *strictly* between 0 and 1.

- □ All rewards and all initial Q values are finite.
- □ The learning rate is constant.
- □ None of the above.

A and C: A is necessary but not sufficient, B is sufficient but not necessary, C is necessary and sufficient, D is neither necessary nor sufficient.

3 Pretraining

1. **Short answer:** In 2-3 concise sentences, describe the relationship between unsupervised pretraining and the autoencoder architecture.

   In unsupervised pretraining, each layer's weights are first learned by treating that layer as the encoder of a 1-layer autoencoder. The weights are learned to minimize the reconstruction error between it's input and output.

2. **Select all that apply:** Which of the following are issues or shortcomings with policy gradient methods such as the likelihood ratio method presented in lecture?

   □ The learned policies must be deterministic.

   □ The number of sampled trajectories needed to estimate the objective function may be very large.

   □ Policy gradient methods are incompatible with contiuous state and action spaces.

   □ Trajectories are sampled from the current policy, which can be suboptimal or unsafe.

   □ None of the above.

   B and D.

3. **Select one:** For a fixed large language model and natural language task, how do few-, one- and zero-shot accuracy tend to relate to one another and how is this affected by model size?

   ○ Few-shot accuracy, one-shot accuracy and zero-shot accuracy are roughly equivalent and stay roughly constant across all model sizes.

   ○ Few-shot accuracy, one-shot accuracy and zero-shot accuracy are roughly equivalent and tend to increase with increasing model size.

   ○ Few-shot accuracy is greater than one-shot, which is greater than zero-shot and the differences are roughly constant across different sized models.

   ○ Few-shot accuracy is greater than one-shot, which is greater than zero-shot and the differences tend to increase with increasing model size.

D

4 Algorithmic Bias

1. **Numerical answer:** Suppose you have binary classification dataset where 20% of the data points have label +1 and the remaining 80% have label −1. What are the precision, recall, and F1-score of a classifier that always predicts +1?

   precision = 0.2, recall = 1, F1 = $\frac{0.4}{1.2} = \frac{1}{3}$

2. **Select all that apply:** Suppose you have a classifier h that is always 100% accurate at some binary classification task. Furthermore, suppose that there is some protected attribute, A, and the percentage of positive labels is *constant* across different values of A. In this setting, which of the following definitions of algorithmic fairness is satisfied by h?

   □ Independence

   □ Separation

   □ Sufficiency

   □ None of the above.

   A, B and C: A is correct because the distribution of labels is the same across different values of A and h will predict exactly that distribution. Similarly, B is correct because conditioned on the label, the distribution of h's prediction is the same across different values of A. Finally, C is also true because h is always correct so the conditional distribution of the label is pure across all different values of A.

3. **Select all that apply:** Suppose you have a classifier h that always predicts a random label for some binary classification task. Furthermore, suppose that there is some protected attribute, A, and the percentage of positive labels varies across different values of A. In this setting, which of the following definitions of algorithmic fairness is satisfied by h?

   □ Independence

   □ Separation

   □ Sufficiency

   □ None of the above.

   A and B: because the classifier's predictions do not depend on any feature or the label, it will always be independent of A. However, the true label and A may not be independent (as the base rates vary across different values of A) so sufficiency may not be satisfied.

5 Learning Theory

1. **Fill in the Blanks:** Complete the following sentence by circling one option in each square:

   In order to prove that the VC-dimension of a hypothesis set $\mathcal{H}$ is D, you must
   show that $\mathcal{H}$ can / cannot shatter any set / some set / multiple sets
   of D data points and can / cannot shatter any set / some set / multiple sets
   of $D+1$ data points.

   In order to prove that the VC-dimension of a hypothesis set $\mathcal{H}$ is D, you must show that $\mathcal{H}$ can shatter some set of D data points and cannot shatter any set of $D+1$ data points.

2. **Math:** For an arbitrary finite hypothesis set $\mathcal{H}$, provide an upper bound on the VC-dimension of $\mathcal{H}$ in terms of $|\mathcal{H}|$.

   $\lceil \log_2 |H| \rceil$: if $d_{\text{VC}}(H) = D$ then $2^D \leq |H|$ so $D \leq \lceil \log_2 |H| \rceil$

3. Let $\mathcal{X} = \mathbb{R}^2$ and $\mathcal{Y} = \{+1, -1\}$. Let

$$H = \{h_{a_1,a_2} \mid a_1, a_2 \in \mathbb{R}^2, h_{a_1,a_2}(x_1, x_2) = +1 \text{ iff } x_1 \leq a_1 \text{ and } x_2 \leq a_2\},$$

   be the hypothesis class corresponding to positive "quarter planes" in $\mathbb{R}^2$

   (a) **Math:** Show that the VC dimension of H is at least 2.

Consider the set of points $S = \{x = (0,1), y = (1,0)\}$. The set of classifiers $h_{0,0}, h_{1,0}, h_{0,1}, h_{1,1}$ label the points x, y as $-1, -1; 1, -1; -1, 1; 1, 1$ respectively. Thus H shatters S and the VC dimension is at least 2. A pictorial solution is also acceptable.

(b) **Math:** Show that the VC dimension of H is at most 2.

Every set of size 3 cannot be shattered by H. Consider any three points $x = (x_1, x_2), y = (y_1, y_2), z = (z_1, z_2) \in \mathbb{R}^2$. Without loss of generality assume that $x_1 \leq y_1$ or $x_1 \leq z_1$, as well as $x_2 \leq y_2$ or $x_2 \leq z_2$ (indeed we can set y, z or just y to be the point(s) with largest co-ordinates). Now, let x be labeled -1, and let y and z be labeled 1. To classify y and z correctly, we must choose a_1 and a_2 such that $y_1 \leq a_1$ and $z_1 \leq a_1$, as well as $y_2 \leq a_2$ and $z_2 \leq a_2$. However, any such a_1, a_2 will misclassify x, so we have shown that no set of size 3 can be shattered by H.

(c) **Math:** Verify the Sauer-Shelah lemma for H by showing that the number of possible distinct labellings produced by H on m points is $O(m^2)$.

For any set of m points, Sauer-Shelah's lemma implies that H can produce $O(m^2)$ distinct labelings. Consider (at most) $m+1$ horizontal and vertical lines each: $2m$ corresponding to the co-ordinates of the points, and two additional lines – a vertical line strictly to the left of all points and a horizontal line strictly below all points. Pairs consisting of one horizontal line $y = a_2$ and one vertical line $x = a_1$ from the set can be used to define $\leq (m+1)^2 = O(m^2)$ hypotheses h_{a_1,a_2} which correspond to all possible unique labelings by hypotheses in H.

6 Ensemble Methods

6.1 Bagging

1. **Short Answers:** Random forests reduce the variance of single decision trees by introducing randomness at different stages of the algorithm: what are the places where we introduce this randomness? For each assertion below, indicate whether it is true or false; if you select false, justify your answer in 1-2 concise sentences.

   (a) **Bootstrap Aggregation:** We choose N random examples without replacement from the dataset every time we train a decision tree in the forest. Doing so ensures that every tree looks at a different set of examples and thus, the forest as a whole will not overfit to the dataset.

   False, we choose them with replacement.

   (b) **Bagging:** Take N random examples with replacement from the dataset every time we train a decision tree in the forest. Doing so and then combining the hypothesis of all trees (by taking majority) reduces variance while still holding the trends and statistical properties of the original dataset.

   True

   (c) **Feature Split Randomization:** Every tree starts with a random subset of features and uses ID3 to split them. This ensures that all trees are not dependent on the same set of features and the forest is robust

   False: Every node uses a random subset to make a spilt.

   (d) **Hypothesis Combination/Aggregation:** We take the majority vote from a random subset of the decision trees at the end. This means that not all decision trees contribute to the final prediction, so the aggregated model is resilient to some trees having high variance.

False: We aggregate results from all the trees. We do not randomly choose N of them to make the final prediction.

2. **Select one:** In an effort to reduce the training error of each individual tree in a random forest, you decide to not perform split-feature randomization and allow every split to consider all of the features. How would you expect this to impact the generalization error of the random forest and why?

   ○ The generalization error will decrease as each individual decision tree will have lower training error.

   ○ The generalization error will decrease as each individual decision tree will make the same splits.

   ○ The generalization error will increase as the depth of each individual decision tree will tend to increase.

   ○ The generalization error will increase as the individual decision trees will become more correlated.

   D

6.2 Boosting

1. You are developing a new boosting algorithm based off of AdaBoost and decide to use the following update rule for the weights:

$$\omega_t^{(i)} = \frac{\omega_{t-1}^{(i)} \alpha_t^{-\frac{1}{2} y^{(i)} h_t(\mathbf{x}^{(i)})}}{Z_t}$$

   (a) **Math:** Derive an expression for the normalization constant, Z_t, as a function of α_t and ϵ_t, the weighted training error.

$$\begin{aligned}
Z_t &= \sum_{i=1}^{n} \omega_{t-1}^{(i)} \alpha_t^{-\frac{1}{2} y^{(i)} h_t(\mathbf{x}^{(i)})} \\
&= \sum_{h_t(\mathbf{x}^{(i)}) \neq y_i} \omega_{t-1}^{(i)} \alpha_t^{\frac{1}{2}} + \sum_{h_t(\mathbf{x}^{(i)}) = y_i} \frac{\omega_{t-1}^{(i)}}{\alpha_t^{\frac{1}{2}}} \\
&= \epsilon_t \alpha_t^{\frac{1}{2}} + \frac{1 - \epsilon_t}{\alpha_t^{\frac{1}{2}}}
\end{aligned}$$

(b) **Math:** Using your answer to part (a), compute the value of α_t that minimizes the normalization constant, Z_t. Express your answer as a function of ϵ_t.

Taking the derivative of Z_t w.r.t. α_t and setting it equal to zero gives:

$$\begin{aligned}
\frac{\partial Z_t}{\partial \alpha_t} &= \frac{\epsilon_t}{2\alpha_t^{\frac{1}{2}}} - \frac{1-\epsilon_t}{2\alpha_t^{\frac{3}{2}}} \tag{1}\\
&\rightarrow \frac{\epsilon_t}{2\hat{\alpha}_t^{\frac{1}{2}}} - \frac{1-\epsilon_t}{2\hat{\alpha}_t^{\frac{3}{2}}} = 0 \tag{2}\\
&\rightarrow \epsilon_t = \frac{1-\epsilon_t}{\hat{\alpha}_t} \tag{3}\\
&\rightarrow \hat{\alpha}_t = \frac{1-\epsilon_t}{\epsilon_t} \tag{4}
\end{aligned}$$

7 SVMs

1. One practical issue that arises in real-world classification tasks is that of imbalanced data, i.e., when one class is far more prevalent than all other classes. Consider, for example, a binary classification task with $y \in \{-1, +1\}$ where 99% of all possible data points have label -1: a simple classifier that always predicts -1 will achieve an accuracy of 99%. This can make it difficult to improve on/assess whether a particular machine learning model has learned anything at all! One way of learning in this setting is to penalize errors on the rare class more than errors on the common class.

   (a) **Math:** You decide to apply SVMs to the setting described above. Formulate a *primal* optimization problem that defines a soft-margin SVM where the soft error on training data points with label $+1$ is penalized 50 times more than the soft error on training data points with label -1 (**Hint:** the constraints associated with your optimization problem should be the same as the one presented in lecture, only the objective function will differ).

$$\text{minimize } \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i:y^{(i)}=-1}\xi^{(i)} + 50C\sum_{i:y^{(i)}=+1}\xi^{(i)}$$
$$\text{subject to } y^{(i)}(\mathbf{w}^T\mathbf{x}^{(i)} + w_0) \geq 1 - \xi^{(i)}\ \forall\ i \in \{1, ..., N\}$$
$$\xi^{(i)} \geq 0\ \forall\ i \in \{1, ..., N\}$$

(b) **Math:** Write out the Lagrangian function and its partial derivatives w.r.t. the optimization variables.

The Lagrangian function $L = L(\mathbf{w}, \mathbf{b}, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta})$ is given by

$$L = \frac{1}{2}||\mathbf{w}||_2^2 + C\sum_{i:y^{(i)}=-1}\xi^{(i)} + 50C\sum_{i:y^{(i)}=+1}\xi^{(i)} + \sum_{i=1}^{N}\alpha_i(1-\xi_i-y^{(i)}(\mathbf{w}^T\mathbf{x}^{(i)}+b)) + \sum_{i=1}^{N}\beta_i(-\xi_i)$$

with constraints on the multipliers

$$\boldsymbol{\alpha}, \boldsymbol{\beta} \geq \mathbf{0}$$

We take the derivatives w.r.t $\mathbf{w}, b, \{\xi_i\}_{i=1}^{N}$to find the optimal primal values:

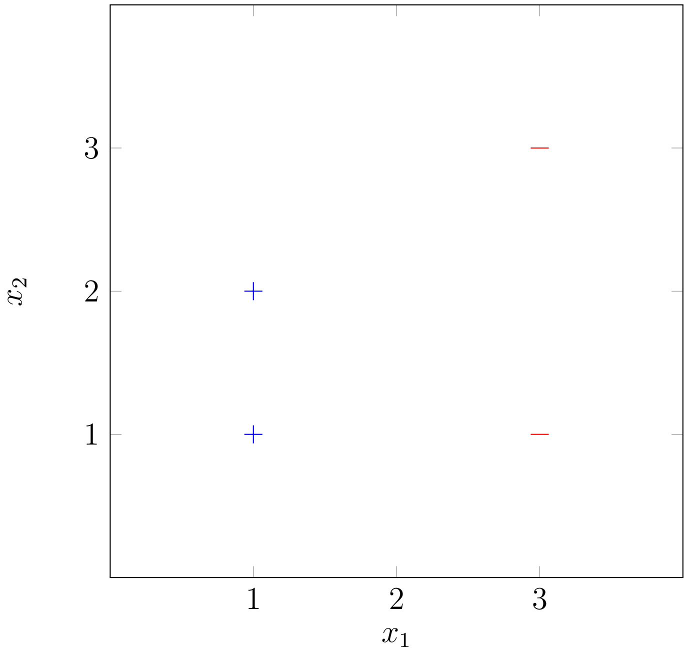
$$\begin{aligned}
\frac{\partial L}{\partial \mathbf{w}} &= \mathbf{w} - \sum_{i=1}^{N} \alpha_i y^{(i)} \mathbf{x}^{(i)} \\
\frac{\partial L}{\partial b} &= -\sum_{i=1}^{N} \alpha_i y^{(i)} \\
\frac{\partial L}{\partial \xi_i} &= C - \alpha_i - \beta_i, && \text{if } y^{(i)} = +1 \\
\frac{\partial L}{\partial \xi_i} &= 50C - \alpha_i - \beta_i, && \text{if } y^{(i)} = -1
\end{aligned}$$

(c) **Math:** Write down the dual of the primal optimization problem in this setting.

Setting the derivatives above to zero, and eliminating β_i we get

$$\begin{aligned}
\text{maximize } & -\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha^{(i)} \alpha^{(j)} y^{(i)} y^{(j)} \mathbf{x}^{(i)} x^{(j)} + \sum_{i=1}^{N} \alpha^{(i)} \\
\text{subject to } & \sum_{i=1}^{N} \alpha^{(i)} y^{(i)} = 0 \\
& 0 \leq \alpha_i \leq C \ \forall \ i : y^{(i)} = -1 \\
& 0 \leq \alpha_i \leq 50C \ \forall \ i : y^{(i)} = +1
\end{aligned}$$

2. Consider the following dataset, consisting of two data points with label + and two data points with label −



   (a) **Drawing:** A non-essential support vector is a support vector that if removed, would not result in a different decision boundary. Conversely, an essential support vector is a support vector that if removed, would change the decision boundary.

   On the figure above, draw the decision boundary you would learn using a hard-margin SVM on this dataset and circle all of the *non-essential* support vectors.

   (b) **Drawing:** Again on the figure provided above, draw a new data point with the label + such that

   - the hard-margin SVM decision boundary changes,
   - the number of essential support vectors *increases* and
   - the number of non-essential support vectors *decreases*.

The original hard-margin SVM decision boundary is $x_1 = 2$. There are three non-essential support vectors: the point at $(1, 1)$, the point at $(1, 2)$, and the point at $(3, 3)$. There are many possible options for the second part but any point with $x_1 \in (1, 3)$ will work.

8 Kernels

1. Car-talk statistician Marge Innovera proposes the following simple kernel function:

$$K(x, x') = \begin{cases} 1 \text{ if } x = x' \\ 0 \text{ otherwise.} \end{cases}$$

(a) **Math:** Prove this is a legal kernel. You may assume the feature space $\mathcal{X}$ is finite. Specifically, describe an implicit mapping $\Phi : \mathcal{X} \to R^m$ (for some value m) such that $K(x, x') = \Phi(x) \cdot \Phi(x')$.

Let Φ be a mapping such that $\Phi(x) \in \mathbb{R}^{|\mathcal{X}|}$. Then there is a component for every element $x' \in \mathcal{X}$. We set that component of $\Phi(x)$ equal to 1 if $x = x'$ and otherwise we set that component equal to 0. Therefore $\Phi(x) \cdot \Phi(x') = 1$ if $x = x'$ and otherwise it equals 0. This means that $K(x, x') = \Phi(x) \cdot \Phi(x')$.

(b) **True or False:** In the Φ-space, any labeling of the points in $\mathcal{X}$ will be linearly separable so we can always run a kernelized version the hard-margin SVM. Briefly justify your answer in 2-3 concise sentences.

○ True

○ False

Let w be a vector in $\mathbb{R}^{|\mathcal{X}|}$. Just like the mapping Φ, there is a component for every element $x \in \mathcal{X}$. We set that component of w equal to 1 if x is in the training set and its label is 1, otherwise we set that component equal to -1. $\Phi(x)$ equals 1 only at the component corresponding to x. Therefore $w \cdot \Phi(x) = 1$ is x is in the training set and its label is 1. Similarly, $w \cdot \Phi(x) = -1$ is x is in the training set and its label is -1. Therefore the training set is linearly separable.

2. **Short answer:** In 2-3 concise sentences, briefly describe the primary benefit of the kernel trick.

   Allows mapping features into higher dimensional space but avoids the extra computational costs of mapping into higher dimensional feature space explicitly.

3. **Select all that apply:** If k is a valid kernel, which of the following statements must be true?

   □ $k(x, x') = k(x', x) \; \forall \; x$ and x'

   □ $k(x, x') \geq 0 \; \forall \; x$ and x'

   □ If $k(x, x) = k(x', x')$, then $x = x'$

   □ $k(x, x') = k(x - x', 0) \; \forall \; x$ and x'

   □ None of the above

   A