CS 475 Machine Learning: Homework 5 Analytical (70 points)

Assigned: Tuesday, Nov. 16th, 2021

Due: Tuesday, Nov. 30th, 2021, 11:59 pm US/Eastern

Partner 1: NAME (JHED), Partner 2: NAME (JHED)

Instructions

We have provided this LaTeX document for turning in this homework. We give you one or more boxes to answer each question. The question to answer for each box will be noted in the title of the box. You can change the size of the box if you need more space.

Other than your name, do not type anything outside the boxes. Leave the rest of the document unchanged.

Do not add text outside of the answer boxes. You are allowed to make boxes larger if needed.

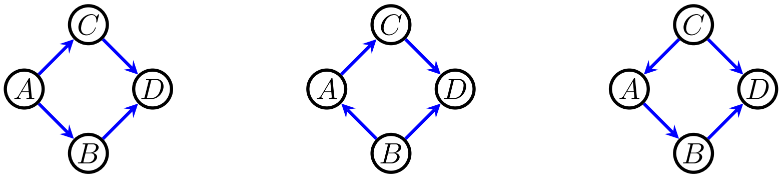
We strongly recommend you review your answers in the generated PDF to ensure they appear correct. We will grade what appears in the answer boxes in the submitted PDF, NOT the original latex file.

Observational Equivalence of DAGs and the GES Algorithm

Question 1.

Two DAGs $\mathcal{G}_1$ and $\mathcal{G}_2$ are said to be *observationally equivalent* if the list of conditional independences implied by the d-separation criterion in both graphs is the same. In other words, $\mathcal{G}_1$ and $\mathcal{G}_2$ are observationally equivalent if they imply the same statistical model.

A result by Thomas Verma and Judea Pearl states that $\mathcal{G}_1$ and $\mathcal{G}_2$ are observationally equivalent if and only if they agree on edge adjacencies (in other words, if V_i and V_j share an edge in $\mathcal{G}_1$, then V_i and V_j share an edge in $\mathcal{G}_2$, and vice versa – ignoring edge orientation), and agree on unshielded colliders. An unshielded collider is a structure of the form $V_i \rightarrow V_k \leftarrow V_j$ such that V_i and V_j do not share an edge. As an example, the following three graphs are observationally equivalent:



The above three DAGs give the same model: $(B \perp\!\!\!\perp C \mid A)$, $(D \perp\!\!\!\perp A \mid B, C)$.

(a) Write out all equivalence classes for DAGs with three vertices. How many equivalence classes are there?

(b) Assume all data is binary. Write down the dimension of each model corresponding to each equivalence class in (a).

(c) Create an undirected graph representing the discrete state space for structure learning, where vertices represent equivalence classes in (a), and there is an edge connecting any two classes where a DAG in one class differs from a DAG in another class by *addition or deletion* precisely one $\rightarrow$ edge.

(d) Assume the GES algorithm performs the following sequence of edge additions and deletions (starting from the empty graph): add $A \rightarrow B$, add $B \rightarrow C$, add $A \rightarrow C$, remove $B \rightarrow C$. Write down all sequences of equivalence classes consistent with this set of edge additions and removals. Note: there could be more than one such sequence.

(e) Consider a DAG $V_1 \rightarrow V_2 \rightarrow V_3 \rightarrow V_4 \rightarrow V_k$. How many DAGs are observationally equivalent to this DAG? Explain.

Missing Data

Question 2.

(a) Consider the following observed data likelihood:

$$\begin{aligned}\mathcal{L}_{[D]}(\beta) = &\prod_{i=1}^{n} \sum_{x_{2i}^{(1)} \text{ if } r_{2i}=0} \sum_{x_{4i}^{(1)} \text{ if } r_{4i}=0} p(x_{1i}, x_{3i}) p(r_{4i} \mid x_{1i}, x_{3i}) p(r_{2i} \mid x_{1i}) p(x_{2i}^{(1)} \mid x_{1i}) p(x_{4i}^{(1)} \mid x_{3i}) \\ &p(x_{2i} \mid r_{2i}, x_{2i}^{(1)}) p(x_{4i} \mid r_{4i}, x_{4i}^{(1)})\end{aligned}$$

Draw the missing data graph for this model.

(b) Does this likelihood represent a missing at random (MAR) model or missing not at random (MNAR) model?

(c) Express $p(x_1, x_2^{(1)}, x_3, x_4^{(1)})$ as a function $p(x_1, x_2, x_3, x_4, r_2, r_4)$.

(d) Consider the following observed data likelihood:

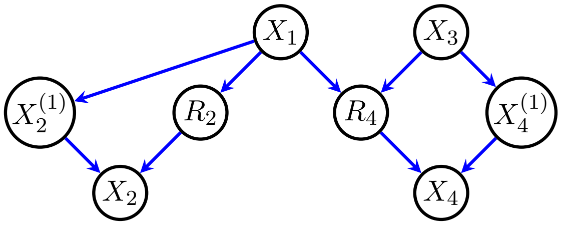
$$\begin{aligned}\mathcal{L}_{[D]}(\beta) = &\prod_{i=1}^{n} \sum_{x_{1i}^{(1)} \text{ if } r_{1i}=0} \sum_{x_{2i}^{(1)} \text{ if } r_{2i}=0} \sum_{x_{3i}^{(1)} \text{ if } r_{3i}=0} p(x_{1i}^{(1)}) p(x_{2i}^{(1)} \mid x_{1i}^{(1)}) p(x_{3i}^{(1)} \mid x_{2i}^{(1)}, x_{1i}^{(1)}) \\ &p(r_{1i} \mid x_{2i}^{(1)}, x_{3i}^{(1)}) p(r_{2i} \mid x_{1i}^{(1)}, r_{3i}) p(r_{3i} \mid x_{2i}^{(1)}, r_{1i}) \\ &p(x_{1i} \mid r_{1i}, x_{1i}^{(1)}) p(x_{2i} \mid r_{2i}, x_{2i}^{(1)}) p(x_{3i} \mid r_{3i}, x_{3i}^{(1)})\end{aligned}$$

Draw the missing data graph for this model.

(e) Does this likelihood represent a missing at random (MAR) model or missing not at random (MNAR) model?

(f) Does the observed data likelihood have a unique global maximum? In other words, is $p(x_1^{(1)}, x_2^{(1)}, x_3^{(1)}, r_1, r_2, r_3)$ a function of the observed data distribution $p(x_1, x_2, x_3, r_1, r_2, r_3)$? Why? (This is a yes/no question with an explanation: if such a function exists, you don't have to give it)

(a) The missing data graph is:

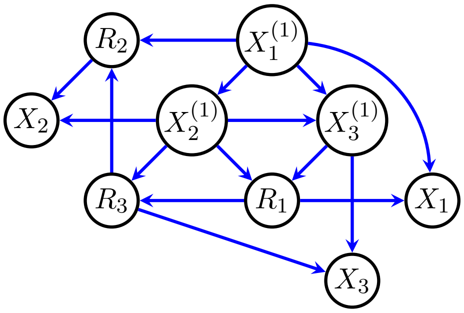


(b) This is a missing at random (MAR) problem. MAR means events that lead to missingness occur independently of unobserved data given observed data, and in this DAG we can clearly see $(X_2^{(1)} \perp\!\!\!\perp R_2 \mid X_1)$ (X_1 fully observed) and $(X_4^{(1)} \perp\!\!\!\perp R_4 \mid X_3)$ (X_3 fully observed). Thus, we fit the definition of MAR.

(c)

$$
\begin{aligned}
p(x_1, x_2^{(1)}, x_3, x_4^{(1)}) =& p(x_1, x_3) p(x_2^{(1)} \mid x_1) p(x_4^{(1)} \mid x_3) \\
=& p(x_1, x_3) p(x_2^{(1)} \mid x_1, r_2 = 1) p(x_4^{(1)} \mid x_3, r4 = 1) \\
=& p(x_1, x_3) p(x_2 \mid x_1, r_2 = 1) p(x_4 \mid x_3, r4 = 1) \\
=& \sum_{x_2, x_4, r_2, r_4} p(x_1, x_2, x_3, x_4, r_2, r_4) \\
& \cdot \frac{p(x_1, x_2, r_2 = 1)}{\sum_{x_2} p(x_1, x_2, r_2 = 1)} \\
& \cdot \frac{p(x_3, x_4, r_4 = 1)}{\sum_{x_4} p(x_3, x_4, r_4 = 1)} \\
=& \sum_{x_2, x_4, r_2, r_4} p(x_1, x_2, x_3, x_4, r_2, r_4) \\
& \cdot \frac{\sum_{x_3, x_4, r_4} p(x_1, x_2, x_3, x_4, r_2 = 1, r_4)}{\sum_{x_2} \sum_{x_3, x_4, r_4} p(x_1, x_2, x_3, x_4, r_2 = 1, r_4)} \\
& \cdot \frac{\sum_{x_1, x_2, r_2} p(x_1, x_2, x_3, x_4, r_2, r_4 = 1)}{\sum_{x_4} \sum_{x_1, x_2, r_2} p(x_1, x_2, x_3, x_4, r_2, r_4 = 1)} \\
=& \sum_{x_2, x_4, r_2, r_4} p(x_1, x_2, x_3, x_4, r_2, r_4) \\
& \cdot \frac{\sum_{x_3, x_4, r_4 \text{ if } r_2 = 1} p(x_1, x_2, x_3, x_4, r_2, r_4)}{\sum_{x_2, x_3, x_4, r_4 \text{ if } r_2 = 1} p(x_1, x_2, x_3, x_4, r_2, r_4)} \\
& \cdot \frac{\sum_{x_1, x_2, r_2 \text{ if } r_4 = 1} p(x_1, x_2, x_3, x_4, r_2, r_4)}{\sum_{x_1, x_2, x_4, r_2 \text{ if } r_4 = 1} p(x_1, x_2, x_3, x_4, r_2, r_4)}
\end{aligned}
$$

(d) The missing data graph is:



(e) This is a missing not at random (MNAR) model. As none of the X are completely known, we can see that $X_1^{(1)}$ and R_1 are not independent because we don't know $X_2^{(1)}$; $X_2^{(1)}$ and R_2 are not independent because we don't know $X_1^{(1)}$; $X_3^{(1)}$ and R_3 are not independent because we don't know $X_1^{(1)}$ and $X_2^{(1)}$. Clearly, this is missing not at random because the missingness and unobserved data are clearly dependent.

(f) no, it does not. we know that all missingness and unobserved data are dependent, so there is no way we can bring $R_i = 1$ condition into the factorization and replace $X_i^{(1)}$ with X_i. Thus, it is not possible write $p(x_1^{(1)}, x_2^{(1)}, x_3^{(1)}, r_1, r_2, r_3)$ as a function of the observed data distribution $p(x_1, x_2, x_3, r_1, r_2, r_3)$.

PCA.

(a) Assume we performed PCA on a centered dataset $[D^*]$ of size n by k, and kept the first m eigenvalues of the sample covariance matrix $\hat{C}$. In other words, the new dataset $[\tilde{D}] = [D^*][V]_m$ is a matrix of size n by m, where $[V]_m$ is a k by m matrix containing the m eigenvectors corresponding to the largest eigenvalues of $\hat{C}$. Let i and j be different column indices in $[V]_m$. Show that the sample covariance between the corresponding columns of $[\tilde{D}]$ is 0 (meaning that the ith and jth features in $[\tilde{D}]$ are uncorrelated.

Hints: the sample covariance of any dataset $[X]$ is proportional to $[X]^T[X]$. You may use the fact that eigenvectors are all orthogonal to each other, meaning that if $\mathbf{v}_i$, $\mathbf{v}_j$ are eigenvectors, then $\mathbf{v}_i\mathbf{v}_j = 0$. Finally, note that given a matrix $[A]$, and its eigenvalue λ_i, and the corresponding eigenvector $\mathbf{v}_i$, $[A]\mathbf{v}_i = \lambda\mathbf{v}_i$.

(b) PCA is a dimension reduction method that aims to find a k-dimensional description of m-dimensional data (where k is hopefully much smaller than m). Kernel PCA aims to project m into a very high dimensional (possibly infinite dimensional) space using a kernel $K(x_i, x_j)$. Given that the goal is to reduce dimension of the original data $[D]$, what is the point of projecting to a high dimensional space prior to reducing dimension to k?

(a) We know that the sample covariance matrix of $[\tilde{D}]$ is proportional to $[\tilde{D}]^T[\tilde{D}]$, so we can first calculate $[\tilde{D}]^T[\tilde{D}]$, and if any element a_{ij} is 0 in $[\tilde{D}]^T[\tilde{D}]$, the covariance of i and j column in $[\tilde{D}]$ should also be 0.

$$\begin{aligned}
[\tilde{D}]^T[\tilde{D}] &= ([D^*][V]_m)^T[D^*][V]_m \\
&= [V]_m^T[D^*]^T[D^*][V]_m
\end{aligned}$$

We know that $\hat{C} = \frac{1}{n-1}[D^*]^T[D^*]$, so we have $[D^*]^T[D^*] = (n-1)\hat{C}$. Plug it in:

$$\begin{aligned}
[\tilde{D}]^T[\tilde{D}] &= [V]_m^T[D^*]^T[D^*][V]_m \\
&= [V]_m^T(n-1)\hat{C}[V]_m \\
&= (n-1)[V]_m^T\hat{C}[V]_m \\
&= (n-1)\begin{bmatrix} \mathbf{v}_1^T \\ \mathbf{v}_2^T \\ \vdots \\ \mathbf{v}_m^T \end{bmatrix} \hat{C} \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_m \end{bmatrix} \\
&= (n-1)\begin{bmatrix} \mathbf{v}_1^T \\ \mathbf{v}_2^T \\ \vdots \\ \mathbf{v}_m^T \end{bmatrix} \begin{bmatrix} \hat{C}\mathbf{v}_1 & \hat{C}\mathbf{v}_2 & \cdots & \hat{C}\mathbf{v}_m \end{bmatrix}
\end{aligned}$$

As $\mathbf{v}_i$ is eigenvector of $\hat{C}$, we have $\hat{C}\mathbf{v}_i = \lambda_i\mathbf{v}_i$:

$$
\begin{aligned}
[\tilde{D}]^T[\tilde{D}] &= (n-1)\begin{bmatrix}\mathbf{v}_1^T \\ \mathbf{v}_2^T \\ \vdots \\ \mathbf{v}_m^T\end{bmatrix}\begin{bmatrix}\hat{C}\mathbf{v}_1 & \hat{C}\mathbf{v}_2 & \cdots & \hat{C}\mathbf{v}_m\end{bmatrix} \\
&= (n-1)\begin{bmatrix}\mathbf{v}_1^T \\ \mathbf{v}_2^T \\ \vdots \\ \mathbf{v}_m^T\end{bmatrix}\begin{bmatrix}\lambda_1\mathbf{v}_1 & \lambda_2\mathbf{v}_2 & \cdots & \lambda_m\mathbf{v}_m\end{bmatrix} \\
&= (n-1)\begin{bmatrix}\lambda_1\mathbf{v}_1^T\mathbf{v}_1 & \lambda_2\mathbf{v}_1^T\mathbf{v}_2 & \cdots & \lambda_m\mathbf{v}_1^T\mathbf{v}_m \\ \lambda_1\mathbf{v}_2^T\mathbf{v}_1 & \lambda_2\mathbf{v}_2^T\mathbf{v}_2 & \cdots & \lambda_m\mathbf{v}_2^T\mathbf{v}_m \\ \vdots & \vdots & \ddots & \vdots \\ \lambda_1\mathbf{v}_m^T\mathbf{v}_1 & \lambda_2\mathbf{v}_m^T\mathbf{v}_2 & \cdots & \lambda_m\mathbf{v}_m^T\mathbf{v}_m\end{bmatrix}
\end{aligned}
$$

We know that eigenvectors are all orthogonal to each other, so $\mathbf{v}_i^T\mathbf{v}_j = 0$ when $i \neq j$. Then to covariance matrix becomes:

$$
\begin{aligned}
[\tilde{D}]^T[\tilde{D}] &= (n-1)\begin{bmatrix}\lambda_1\mathbf{v}_1^T\mathbf{v}_1 & \lambda_2\mathbf{v}_1^T\mathbf{v}_2 & \cdots & \lambda_m\mathbf{v}_1^T\mathbf{v}_m \\ \lambda_1\mathbf{v}_2^T\mathbf{v}_1 & \lambda_2\mathbf{v}_2^T\mathbf{v}_2 & \cdots & \lambda_m\mathbf{v}_2^T\mathbf{v}_m \\ \vdots & \vdots & \ddots & \vdots \\ \lambda_1\mathbf{v}_m^T\mathbf{v}_1 & \lambda_2\mathbf{v}_m^T\mathbf{v}_2 & \cdots & \lambda_m\mathbf{v}_m^T\mathbf{v}_m\end{bmatrix} \\
&= (n-1)\begin{bmatrix}\lambda_1\mathbf{v}_1^T\mathbf{v}_1 & 0 & \cdots & 0 \\ 0 & \lambda_2\mathbf{v}_2^T\mathbf{v}_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_m\mathbf{v}_m^T\mathbf{v}_m\end{bmatrix}
\end{aligned}
$$

We can see that except for the main diagonal elements which are proportional to the sample covariance between the same column and itself in $[\tilde{D}]$, all other elements are zero. The zeros in covariances matrix of $[\tilde{D}]$ show that any two different columns of $[\tilde{D}]$ has covariance zero. This means that for all different columns in $[V]_m$, the sample covariance between the corresponding columns of $[\tilde{D}]$ are zero, as there is one-to-one correspondence between columns in $[V]_m$ and $[\tilde{D}]$.

(b) There are several reasons and benefits for using kernel PCA. First, PCA itself is a linear transformation, and each principal component is a linear combination of the original data point. That means when doing PCA, we are actually assuming linearity in the data, and are trying to use linear transformation to map the data into lower dimensions with largest variance. However, for many datasets, especially the high-dimensional data, it is not possible to find a lower dimensional space using linear transformation that can separate the data well (i.e. find eigenvector with large variance associated with it). In this case, the separation of data is non-linear, and we would need a non-linear transformation to map the data into a better low-dimensional space with larger variance in each axis. This is why we need a kernel here. The kernel trick first use a non-linear transformation to transform the data into a much higher dimensional space, and then finding eigenvector there can give you a larger variance and better separation of data in the resulting low-dimensional space. Second, even if we have transformed the data to a higher dimensional space and reduced it again, the non-linear transformation we added through this process will likely to cause the data to lie on a lower dimensional subspace of it. Thus, we increased the dimension first in order to be able to better decrease it into a lower dimensional space with higher variance and better separation of data.