**Homework**

**Bayesian Networks (CDA 500)**

**Question 1:**  
You are provided with a dataset, “disease\_data.csv”, which contains information about patients, fever, cough, sore throat, results for a test, and diagnosis.

**a)** Load and clean the data. Ensure all variables are converted to factors. Remove the Patient\_ID column.

**B)** Use **hill-climbing** (score-based) and **Grow-Shrink** (constraint-based/ independence tests) to learn two network structures. Visualize both.

* Which edges are consistent across both methods?
* Which ones differ, and why might that be?
* Plot the CPDAGs, are they equivalent?

**C)** Choose your preferred structure from part A (or manually modify it) and **fit the conditional probability tables (CPTs)** using bn.fit().

**D)** Use the fitted network to answer the following probabilistic queries:

1. What is the probability that a patient has the **Flu** given that they have a **Fever** and **Test\_Result = Positive**? Use both exact inference and approximate inference. Compare them for computational time, accuracy, and suitability in real-world settings.
2. What is the most likely diagnosis for a patient with **Cough = Yes, Sore\_Throat = Yes,** and **Test\_Result = Negative**? Use both exact inference and approximate inference. Compare them for computational time, accuracy, and suitability in real-world settings.

**E)** Generate a synthetic dataset of 200 samples from your fitted model. Fit a new structure. How does it compare to the original structure learned from data?

**F)** Generate a synthetic dataset of 750 samples from your fitted model. Fit a new structure. How does it compare to the original structure learned from data?

**G)** Suggest at least one real-world application of this approach (e.g., triaging, testing strategy) and describe how this model could be updated with new data over time.

**Question 2**You are given a flight network between six cities: A, B, C, D, E, F.  
Edges represent direct flight paths:

A → B, A → C  
B → C, B → D  
C → E  
D → E, D → F  
E → F

a) Model this as a **Bayesian Network**, where each node is a binary variable indicating whether a **delay** occurred.

b) Assign conditional probabilities (choose reasonable values based on real-world intuition).

c) Use bnlearn’s “model2network()” and gRain’s “compile()” to compute the probability of a delay at F, given a delay at A.

d) Compare this with the estimate from **likelihood-weighted sampling** using cpquery. Report both results and comment on when exact vs approximate inference is preferable.

**Question 3**

Consider the network in Figure 4.1 in the bottom left panel (Scutari, Bayesian Networks with examples in R).

1. Extend the network in the bottom-left panel to model as a second- time point t2 in addition to t0 and t1. Call the new nodes St2, Tin and Tout. How many new parameters does that require?
2. Create a BN object encoding your new network in part A. Use the conditional probabilities from Section 4.3 to create the bn.fit object.
3. Using probabilistic reasoning, use “cpquery” to compute the probability that Tin2 is equal to “18-24” and St2 is “low” given that Tin0 is “18-24” and St0 is “high” when the windows are either open or closed. What would you expect from a similar performance as in Section 4.5?

**Question 4**   
You are given a simulated social network of 30 individuals. Each node represents a person, and each edge represents a mutual friendship. You may simulate a scale-free network using the preferential attachment model:

library(igraph)

set.seed(123)

g <- sample\_pa(n = 30, power = 1, m = 2, directed = FALSE)

1. Plot the network using a force-directed layout (layout\_with\_fr()). Label nodes with unique IDs.
2. Compute and report the following: Number of nodes and edges, graph density, graph diameter, and whether the graph is connected.
3. Compute the following centrality measures for each node:  
   - Degree centrality  
   - Closeness centrality  
   - Betweenness centrality  
   Report the top 3 nodes for each measure. What do these centralities tell you about influence or information access in the network?
4. Identify all maximal cliques using cliques() and largest\_cliques(). How many are there? What is the size of the largest?
5. Use Jaccard similarity (similarity(g, method = "jaccard")) to identify the top 3 most likely future friendships (i.e., edges not currently in the graph with high similarity). Explain your results briefly.
6. Compute the shortest path between node 5 and node 20 using shortest\_paths().
7. Use at least one community detection algorithm (cluster\_louvain, cluster\_walktrap, etc.) to:  
   Assign community membership to each node.  
   Visualize the communities on the network plot from part A.  
   Report the modularity score of the detected community structure.
8. Identify any bridge nodes (nodes that connect different communities) using betweenness centrality or inspection of inter-community edges.