# Diabetes Prediction with Incomplete Patient Data

Hao Yi Ong, Dennis Wang, Xiao Song Mu

*CS 221 Artificial Intelligence: Principles and Techniques Class Project*

## Introduction

- Given a set of electronic health records, we want to have a smart predictor that prompts high-risk patients to obtain Type II Diabetes testing



- Original Kaggle challenge:
  - Patients all have a standard database and a full medical record
  - I.e., exact same tests taken, same variables recorded, etc.

- In practice, however:
  - Not everyone has taken the same tests and gotten regular checkups
  - Database inconsistencies or errors in inputting data may exist

- Predictor must be able to accurately classify based on incomplete or erroneous medical records to be useful

## Diabetes Prediction

- Given
  - **Training set** containing standardized patient medical records
  - **Testing set** containing patient medical records with missing information and erroneously recorded data
- Output
  - **Bayesian network structure** encoding the conditional dependencies between medical record variables
  - **Bayesian network parameters** encoding the conditional probabilities for each variable
  - **Probabilistic inference** on the learned Bayesian network (BN) for classification
- To minimize the error rate, including false positives and false negatives, on classifying whether a patient has Type II Diabetes

## Evaluation Criteria
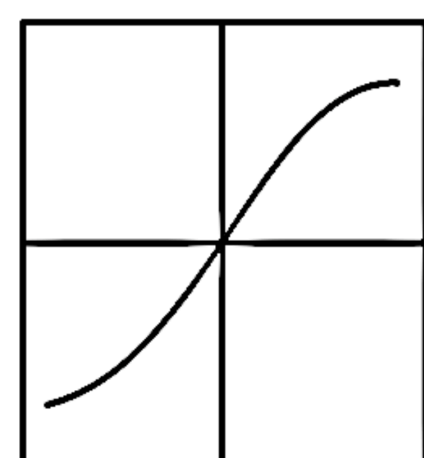


- **Baseline: logistic regression**
  - Basic features: age, BMI, ...
  - 10% hold-out cross validation
  - False positive rate of 0.7%, false negative rate of 15.3%, and an error rate of 16% (84% accuracy)
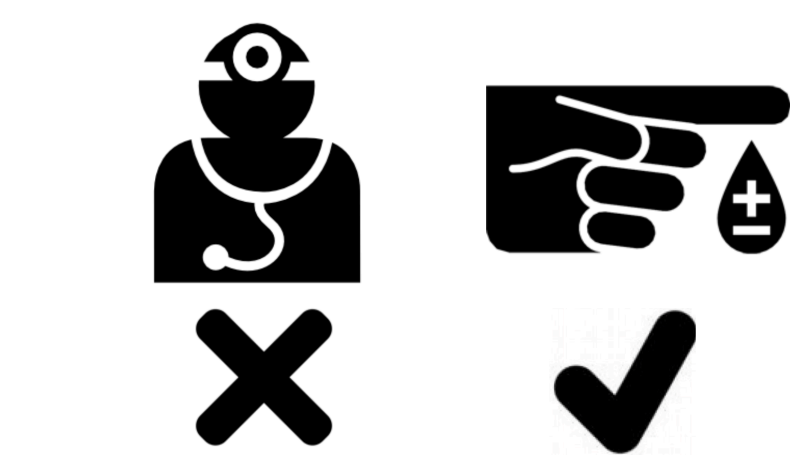
- **Oracle**
  - Ideal oracle: experienced physicians (impractical)
  - Use as surrogate diabetes tests vetted by HHS (HBA1c, FPG, and OGTT; 85–95% accuracy)

## Structure Learning as a Search Problem

- **NP-hard search problem** Finding the best set of edges is hard as the number of BNs (DAGs) grows superexponentially with the number of nodes
- **Bayesian score** To evaluate the BN, we use the Bayesian score with a uniform Dirichlet prior over structures, which optimally balances the complexity of the BN structure with the available data (Koller and Friedman, 2009)
- **Feature selection** We selected 27 discretized features (nodes) for the 10,000 samples according to ICD9 disease groupings to limit the search space
- **Tabu search** To find the optimal structure, we use a hill-climbing algorithm that maximizes the "fitness" of the BN based on the Bayesian score

  By maintaining a tabu list of recent operators we applied (e.g., adding an edge to the existing BN) and not considering operators that reverse the effect of recently applied operators, the heuristic avoids getting stuck at local optima

  **given** training set $\mathcal{D}$, structure prior $P(\mathcal{G})$, node set $\mathcal{N}$, tabu list size $L$
  **initialize** random BN $\mathcal{G}$, tabu list $\mathcal{T}$, valid operations $\mathcal{O}$
  **repeat**
      1. if $|\mathcal{O}|$ too large, generate random subset of valid operations $\hat{\mathcal{O}} \subset \mathcal{O}$
      2. find best operation: $\text{Op} := \text{argmax}_{\text{Op} \in \hat{\mathcal{O}} \backslash \mathcal{T}} \text{BayesScore}(\text{Op}(\mathcal{G}))$
      3. set $\mathcal{G} := \text{Op}(\mathcal{G})$ and $\mathcal{T} := \mathcal{T} \cup \{\text{reverse}(\text{Op})\}$
      4. remove operation added $L$ iterations ago to $\mathcal{T}$ from it
  **until** BayesScore($\mathcal{G}$) converges
  **return** $\mathcal{G}$

- **Resulting structures** Produced 20 structures; avg 31 min with Julia implementation on an Intel Xeon E5-2650 processor (2.60 GHz), 32 GB RAM

## Parameter Learning

- **MLE with Laplace smoothing** We use MLE with Laplace smoothing to learn the conditional probabilities associated with each node (9,000 samples)
- **Limited dataset** Given dataset is small in comparison to number of possible variable values, which motivates our use of Laplace smoothing
- **Relational database** We store our data in a relational database and optimize data retrieval by applying indices over sets of variable names

## Approximate Probabilistic Inference

- **Classification using BN** Assign label to whichever inferred probability is higher; probabilities are over the missing variables and observed ones
- **NP-hard** Given our large BN structure and variable domain sizes, exact inference is infeasible (exponential time complexity in the worst case)
- **Gibbs sampling** We use a Markov Chain Monte Carlo technique where, in the limit, samples are drawn exactly from the joint distribution over the unknown variables given the observed variables

  Since samples are not independent, we discard the first 100 samples (burn-in period) and keep only every 10th sample (thinning)

## Results and Analysis

- **Best structure** We picked the top 8 structures based on the Bayesian score, ran parameter learning on them, and tested their prediction accuracies based on classifying test sets using inference
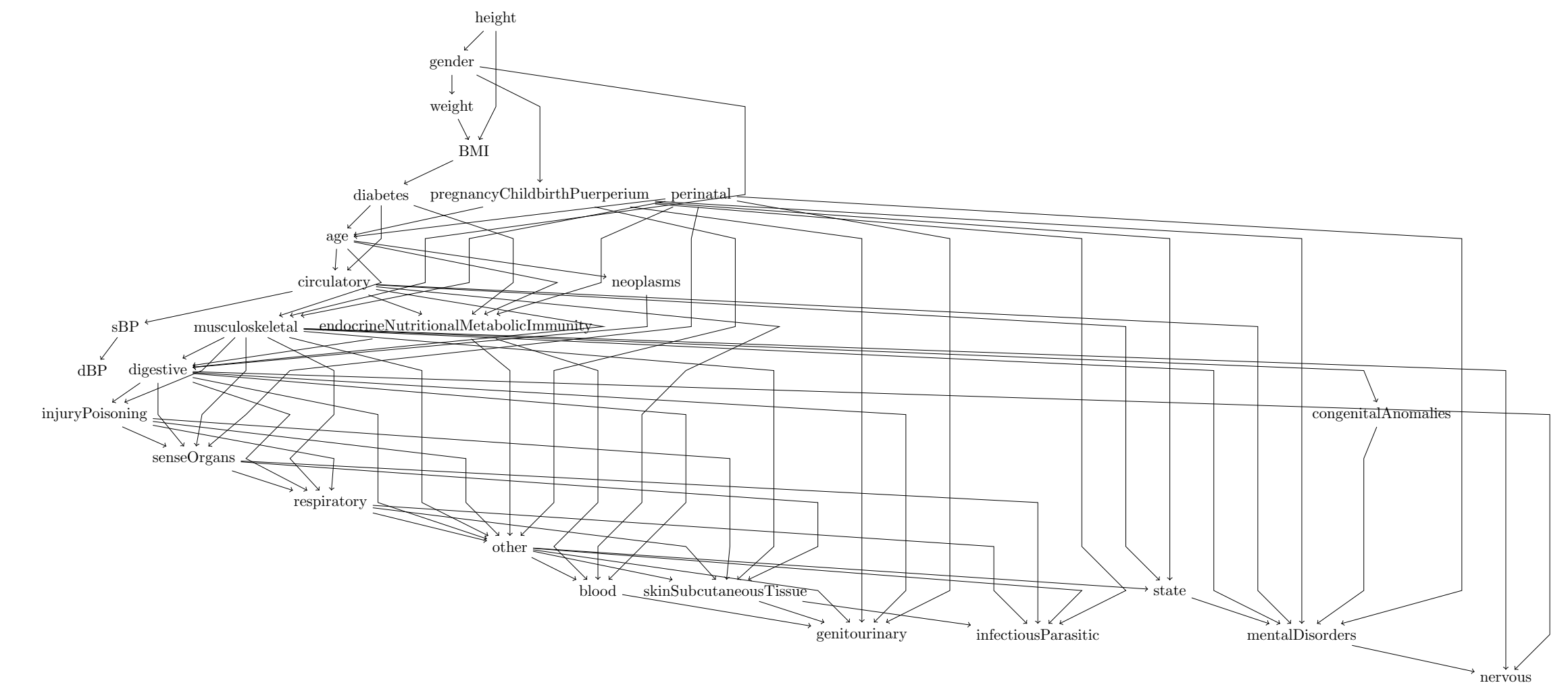


- **Error rates** Varying the number of missing variables, the error rates were 15–19% on 1,000 test samples, which are competitive with the 16% baseline
- **Runtime** Depending on the number of missing variables, a single prediction can take up to 30 seconds—much more expensive than our baseline
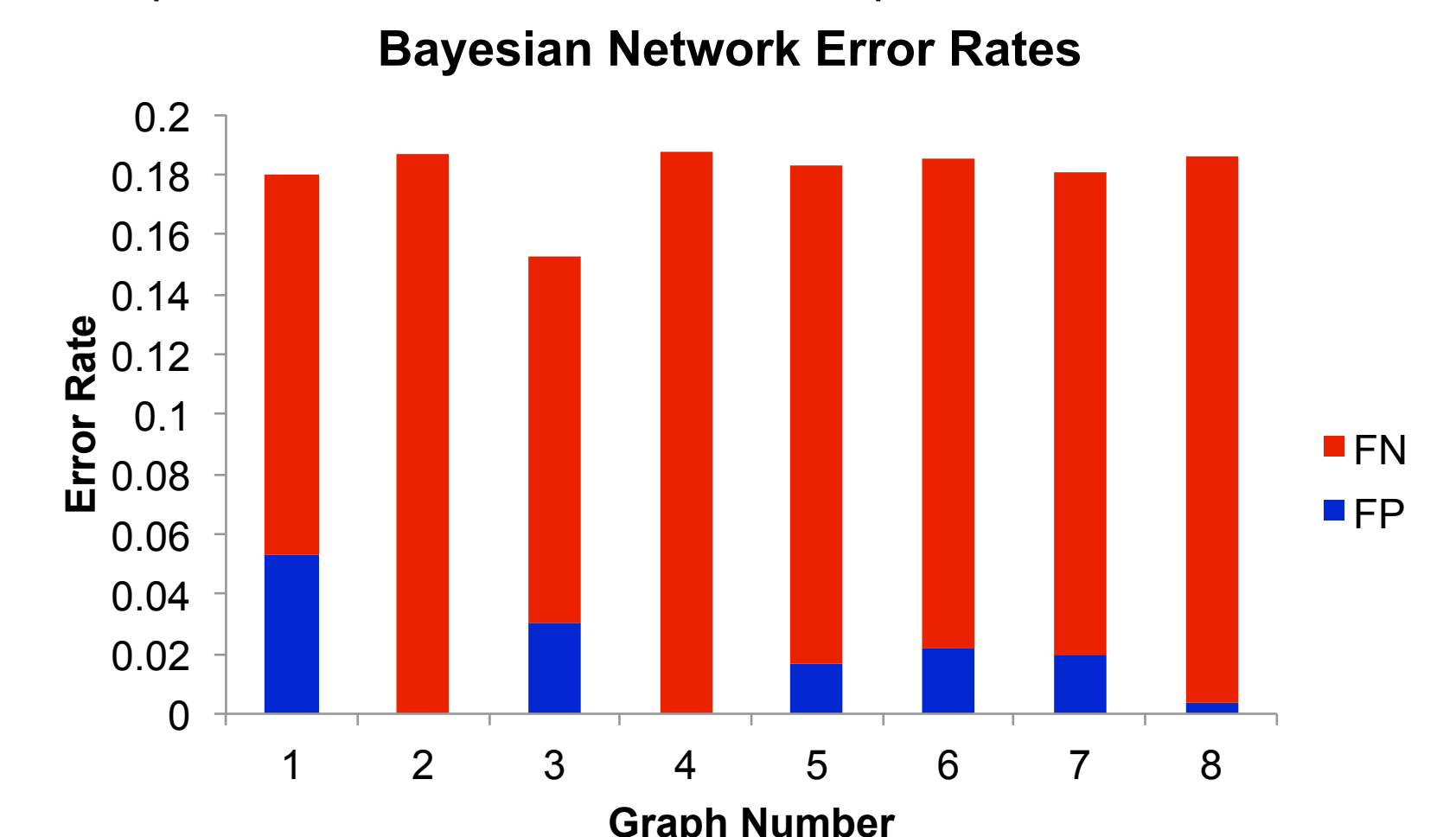


Summary: Classification results with varying number of missing features

| #missing | Error rate | False positive | False negative | Pred time (sec) |
|---|---|---|---|---|
| 3 | 0.1662 | 0.0363 | 0.1299 | 3.754 |
| 5 | 0.1511 | 0.0302 | 0.1208 | 6.437 |
| 8 | 0.1339 | 0.0272 | 0.1067 | 9.672 |
| 10 | 0.1299 | 0.0252 | 0.1047 | 11.859 |

## Conclusion and Extensions

- Our model is suitable for diabetes prediction on existing databases (even with incomplete records) to detect patients at risk for diabetes
- Incorporate expert knowledge in feature selection and deciding the structure
- Consider different inference methods for runtime and accuracy improvement