Using Past to Predict Future – Bayesian Networks
and Medical Data

Jaroslaw ‘Jaric’ Zola

jzola@buffalo.edu · http://www.score-group.org/

SCoRe – Scalable Computing Research Group

Department of Computer Science and Engineering
Department of Biomedical Informatics
University at Buffalo, SUNY
• Suppose you are taking “Statistics” this semester, what are your chances of getting “A”? $P(Stat = A) = \ldots$
Simple Exercise

- Suppose you are taking “Statistics” this semester, what are your chances of getting “A”?  \( P(\text{Stat} = A) = \ldots \)

- Which factors are important to answer this question?
Simple Exercise

- How much effort you will put into the course, $Eff$
- Your grade in math, $Math$
- Your overall workload, $Work$
- How much you like science $Sci$
Simple Exercise

- Intuitively, we could plot dependencies between our variables as follows:
Simple Exercise

- Intuitively, we could plot dependencies between our variables as follows:

```
<table>
<thead>
<tr>
<th>Sci</th>
<th>Like</th>
<th>Dislike</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.75</td>
<td>0.25</td>
</tr>
</tbody>
</table>
```

```
<table>
<thead>
<tr>
<th>Sci \ Math</th>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Like</td>
<td>0.50</td>
<td>0.25</td>
<td>0.25</td>
</tr>
<tr>
<td>Dislike</td>
<td>0.10</td>
<td>0.40</td>
<td>0.50</td>
</tr>
</tbody>
</table>
```

```
<table>
<thead>
<tr>
<th>Work \ Low</th>
<th>High</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.4</td>
<td>0.6</td>
</tr>
</tbody>
</table>
```

```
<table>
<thead>
<tr>
<th>Math \ Eff</th>
<th>Low</th>
<th>High</th>
</tr>
</thead>
<tbody>
<tr>
<td>Like Low</td>
<td>0.15</td>
<td>0.85</td>
</tr>
<tr>
<td>Like High</td>
<td>0.25</td>
<td>0.75</td>
</tr>
<tr>
<td>Dislike Low</td>
<td>0.50</td>
<td>0.50</td>
</tr>
<tr>
<td>Dislike High</td>
<td>0.40</td>
<td>0.60</td>
</tr>
</tbody>
</table>
```

```
<table>
<thead>
<tr>
<th>Stat \ Math</th>
<th>Fail</th>
</tr>
</thead>
<tbody>
<tr>
<td>A Low</td>
<td>0</td>
</tr>
<tr>
<td>A High</td>
<td>0.5</td>
</tr>
<tr>
<td>B Low</td>
<td>0.10</td>
</tr>
<tr>
<td>B High</td>
<td>0.15</td>
</tr>
<tr>
<td>C Low</td>
<td>0.15</td>
</tr>
<tr>
<td>C High</td>
<td>0.01</td>
</tr>
</tbody>
</table>
```

```
<table>
<thead>
<tr>
<th>Eff \ Low</th>
<th>High</th>
</tr>
</thead>
<tbody>
<tr>
<td>A Low</td>
<td>0</td>
</tr>
<tr>
<td>A High</td>
<td>0.85</td>
</tr>
<tr>
<td>B Low</td>
<td>0.10</td>
</tr>
<tr>
<td>B High</td>
<td>0.15</td>
</tr>
<tr>
<td>C Low</td>
<td>0.15</td>
</tr>
<tr>
<td>C High</td>
<td>0.01</td>
</tr>
</tbody>
</table>
```
Simple Exercise

- We can describe our joint probability as:

\[
What Are Bayesian Networks?

- Class of Probabilistic Graphical Models
- Efficient and intuitive way to encode conditional independencies
- Formally: \((G, P)\) where \(G = (\mathcal{X}, E)\) is a DAG of conditional independencies and \(P\) is a probability over \(\mathcal{X}\)
• Suppose that \( \mathbf{X} = \{X_1, \ldots, X_n\} \)

• From the chain rule of probability:

\[
P(X_1, \ldots, X_n) = P(X_1)P(X_2|X_1)P(X_3|X_2, X_1) \ldots P(X_n|X_{n-1}, \ldots, X_1)
\]

• BN \((G = (\mathbf{X}, E), P)\) provides much more efficient factorization:

\[
P(X_1, \ldots, X_n) = \prod_{i=1}^{n} P(X_i|Pa(X_i))
\]

where \(Pa(X_i)\) are parents of \(X_i\) in \(G\)
What is $P(\text{Survival} = \text{yes}|\text{Stage} = 0)$?
Which treatment to choose?

$$\arg\max_t P(Survival = yes | Evidence, Treatment = t)$$
Power of Bayesian Networks

Which probabilities are needed to answer queries of interest?
Real Networks Are Complicated
Important Questions

- Where all these probabilities come from?
- How do we build our network?
Bayesian Networks Workflow

Large Data $\xrightarrow{structure \ learning}$ Network Structure $\xrightarrow{inference}$ Predictions

$\xrightarrow{parameter \ learning}$ Conditional Probabilities
Where Probabilities Come From?

Data!
Structure Learning

- Structure is a graph that best explains our data
  \[
  \text{Score}(G) = P(G|D)
  \]

\[
\text{Score}(G) = \frac{P(D|G)P(G)}{P(D)}
\]

- We want to find a graph with the highest Score
Structure Learning

- How many graphs (DAGs) with $n$ variables?
Structure Learning

- How many graphs (DAGs) with $n$ variables?

\[
\begin{array}{c|c}
 n & y \\
 \hline
 1 & 1 \\
\end{array}
\]
Structure Learning

- How many graphs (DAGs) with $n$ variables?

<table>
<thead>
<tr>
<th>$n$</th>
<th>$y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>
Structure Learning

- How many graphs (DAGs) with \( n \) variables?

<table>
<thead>
<tr>
<th>( n )</th>
<th>( y )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>
Structure Learning

- How many graphs (DAGs) with $n$ variables?

<table>
<thead>
<tr>
<th>$n$</th>
<th>$y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>25</td>
</tr>
<tr>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>
Structure Learning

- How many graphs (DAGs) with $n$ variables?

<table>
<thead>
<tr>
<th>$n$</th>
<th>$y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>25</td>
</tr>
<tr>
<td>4</td>
<td>543</td>
</tr>
<tr>
<td>5</td>
<td>29,281</td>
</tr>
<tr>
<td>6</td>
<td>3,781,503</td>
</tr>
</tbody>
</table>

$$y(n) = \sum_{i=1}^{n} (-1)^{i+1} \binom{n}{i} 2^{i(n-i)} y(n-i)$$
Structure Learning

- Search space grows super-exponentially, and our problem is $NP$-hard
- If we can decompose $Score$ as
  \[
  Score(G) = \sum_{i=1}^{n} s(X_i, Pa(X_i))
  \]
  then:
  1. We can use DAG to order our variables
  2. We can disregard ordering of parents
  3. This reduces the search space to $2^n$

- We still need a better approach!
Modern Computers Are Parallel

- 1993: Connection Machine (CM-5), $50,000,000
  1024 cores, 130 Gflop/s

- 2015: Intel Core i7, $1,000-$2,000
  4-8 cores, 80-160 Gflop/s
Modern Computers Are Parallel

- 1993: Connection Machine (CM-5), $50,000,000
  1024 cores, 130 Gflop/s

- 2015: Intel Core i7, $1,000-$2,000
  4-8 cores, 80-160 Gflop/s
The Joy of Parallel Computing
Parallel Structure Learning

- We consider all subsets $A \subseteq \{X_1, \ldots, X_n\}$ in increasing size
- For $A$ we find best parents of $X_i$ from $A - \{X_i\}$
Parallel Structure Learning

- Data with $m = 500$ observations

![Graph showing time in seconds vs. number of processors]
Applications of Bayesian Networks

- Clinical decision/support systems
- Gene networks and genes epistasis
- Recommender systems
- Diagnostic systems
Applications of Bayesian Networks

- Clinical decision/support systems
- Gene networks and genes epistasis
- Recommender systems
- Diagnostic systems
- And Microsoft clippy...

IRRITATION LEVEL

LEGENDARY
Things to Remember

- By learning from data we can make predictions about most likely outcomes
- Bayesian networks help to organize and use joint probabilities
- Parallel computing helps to tackle intractable problems
Questions?

http://www.score-group.org/
http://www.jzola.org/