Wissensentdeckung in Datenbanken

LBP, Strukturlernen

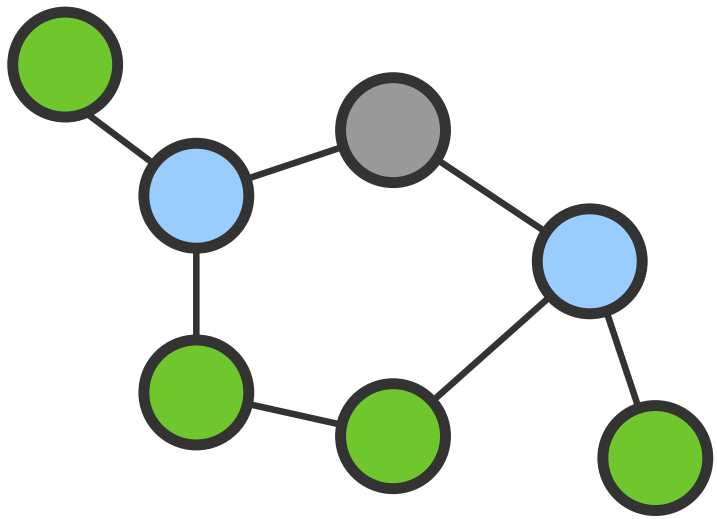
Nico Piatkowski und Uwe Ligges

Informatik—Künstliche Intelligenz
Computergestützte Statistik
Technische Universität Dortmund

04.07.2017

Überblick

- Wiederholung: Belief Propagation
  - Berechnung der Randverteilung
- Strukturlernen—“Woher kommt der Graph?”
  - Was bedeutet der Graph?
  - Chow-Liu Bäumen
  - Regularisierung
- Merkmalsauswahl
  - Forward-Selection
  - Regularisierung



Rückblick (I)

(1) Die Zielfunktion von probabilistischen Modellen ist die (mittlere) negative Log-Likelihood (02.05.)

(2) Die Parameter β des graphischen Modells lernen wir mittels Gradientenabstieg (04.05. + Ü)

(3) Der Gradient besteht aus partiellen Ableitungen (04.05.)

(4) Im graphischen Modell entspricht eine partielle Ableitung bezüglich Parameter β_i dem Erwartungswert der i-ten Komponente der Zufallsvariable $\phi(\boldsymbol{X})$ (29.06.)

Rückblick (I)

(1) Die Zielfunktion von probabilistischen Modellen ist die (mittlere) negative Log-Likelihood (02.05.)

(2) Die Parameter $\boldsymbol{\beta}$ des graphischen Modells lernen wir mittels Gradientenabstieg (04.05. + Ü)

(3) Der Gradient besteht aus partiellen Ableitungen (04.05.)

(4) Im graphischen Modell entspricht eine partielle Ableitung bezüglich Parameter β_i dem Erwartungswert der i-ten Komponente der Zufallsvariable $\phi(\boldsymbol{X})$ (29.06.)

Rückblick (I)

(1) Die Zielfunktion von probabilistischen Modellen ist die (mittlere) negative Log-Likelihood (02.05.)

(2) Die Parameter $\boldsymbol{\beta}$ des graphischen Modells lernen wir mittels Gradientenabstieg (04.05. + Ü)

(3) Der Gradient besteht aus partiellen Ableitungen (04.05.)

(4) Im graphischen Modell entspricht eine partielle Ableitung bezüglich Parameter β_i dem Erwartungswert der i-ten Komponente der Zufallsvariable $\phi(\boldsymbol{X})$ (29.06.)

Rückblick (I)

(1) Die Zielfunktion von probabilistischen Modellen ist die (mittlere) negative Log-Likelihood (02.05.)

(2) Die Parameter $\boldsymbol{\beta}$ des graphischen Modells lernen wir mittels Gradientenabstieg (04.05. + Ü)

(3) Der Gradient besteht aus partiellen Ableitungen (04.05.)

(4) Im graphischen Modell entspricht eine partielle Ableitung bezüglich Parameter $\boldsymbol{\beta}_i$ dem Erwartungswert der i-ten Komponente der Zufallsvariable $\phi(\boldsymbol{X})$ (29.06.)

Rückblick (II)

(5) Bei diskreten Modellen ist $\phi(\boldsymbol{X})_i$ binär (22.06. ff.) und ihr Erwartungswert entspricht der Wahrscheinlichkeit $\mathbb{P}(\phi(\boldsymbol{X})_i = 1)$ (u.a. 29.06.)

(6) Jede Komponente von $\phi(\boldsymbol{X})$ entspricht einem Ereignis $\boldsymbol{X}_C = \boldsymbol{y}$, d.h. $\mathbb{P}(\phi(\boldsymbol{X})_i = 1) = \mathbb{P}(\boldsymbol{X}_C = \boldsymbol{y})$ mit $\boldsymbol{y} \in \mathcal{X}_C$

$\Rightarrow$ Wir brauchen einen Algorithmus zur Berechnung von $\mathbb{P}(\boldsymbol{X}_C = \boldsymbol{y})$!

Rückblick (II)

(5) Bei diskreten Modellen ist $\phi(\boldsymbol{X})_i$ binär (22.06. ff.) und ihr Erwartungswert entspricht der Wahrscheinlichkeit $\mathbb{P}(\phi(\boldsymbol{X})_i = 1)$ (u.a. 29.06.)

(6) Jede Komponente von $\phi(\boldsymbol{X})$ entspricht einem Ereignis $\boldsymbol{X}_C = \boldsymbol{y}$, d.h. $\mathbb{P}(\phi(\boldsymbol{X})_i = 1) = \mathbb{P}(\boldsymbol{X}_C = \boldsymbol{y})$ mit $\boldsymbol{y} \in \mathcal{X}_C$

$\Rightarrow$ Wir brauchen einen Algorithmus zur Berechnung von $\mathbb{P}(\boldsymbol{X}_C = \boldsymbol{y})$!

Rückblick (II)

(5) Bei diskreten Modellen ist $\phi(\boldsymbol{X})_i$ binär (22.06. ff.) und ihr Erwartungswert entspricht der Wahrscheinlichkeit $\mathbb{P}(\phi(\boldsymbol{X})_i = 1)$ (u.a. 29.06.)

(6) Jede Komponente von $\phi(\boldsymbol{X})$ entspricht einem Ereignis $\boldsymbol{X}_C = \boldsymbol{y}$, d.h. $\mathbb{P}(\phi(\boldsymbol{X})_i = 1) = \mathbb{P}(\boldsymbol{X}_C = \boldsymbol{y})$ mit $\boldsymbol{y} \in \mathcal{X}_C$

$\Rightarrow$ Wir brauchen einen Algorithmus zur Berechnung von $\mathbb{P}(\boldsymbol{X}_C = \boldsymbol{y})$!

Loopy Belief Propagation

- Hier: Einschränkung auf Graphen mit $\max_{C \in \mathcal{C}(G)} |C| = 2$
- Dann: Menge der Cliquen $\mathcal{C}(G)$ = Menge der Kanten E
- Der Gradient enthält dann $\mathbb{P}(\boldsymbol{X}_{\{v,u\}} = \boldsymbol{y})$ für alle $\{v, u\} \in E$ und alle $\boldsymbol{y} \in \mathcal{X}_{\{v,u\}}$

Algorithmus:

(1) Lege zwei Arrays m^{neu} und m^{alt} an (jeweils $\sum_{\{v,u\} \in E} (|\mathcal{X}|_v + |\mathcal{X}|_u)$ Einträge) und initialisiere mit 1

(2) Berechne für alle Kanten $\{v, u\}$ in E, beide Kantenrichtungen $s \to t$ (d.h. $v \to u$ und $u \to v$) und jeden Zustand $x \in \mathcal{X}_t$:

$$m^{\text{neu}}_{v \to u}(x) = \sum_{y \in \mathcal{X}_v} \exp(\langle \boldsymbol{\beta}_{uv}, \phi_{uv}(x, y) \rangle) \prod_{w \in \mathcal{N}(v) \setminus \{u\}} m^{\text{alt}}_{w \to v}(y)$$

(3) Falls $\|m^{\text{neu}} - m^{\text{alt}}\| > \varepsilon$: $m^{\text{alt}} \leftarrow m^{\text{neu}}$ und GOTO (2)

Loopy Belief Propagation

- Hier: Einschränkung auf Graphen mit $\max_{C \in \mathcal{C}(G)} |C| = 2$
- Dann: Menge der Cliquen $\mathcal{C}(G)$ = Menge der Kanten E
- Der Gradient enthält dann $\mathbb{P}(\boldsymbol{X}_{\{v,u\}} = \boldsymbol{y})$ für alle $\{v, u\} \in E$ und alle $\boldsymbol{y} \in \mathcal{X}_{\{v,u\}}$

Algorithmus:

(1) Lege zwei Arrays m^{neu} und m^{alt} an (jeweils $\sum_{\{v,u\} \in E} (|\mathcal{X}|_v + |\mathcal{X}|_u)$ Einträge) und initialisiere mit 1

(2) Berechne für alle Kanten $\{v, u\}$ in E, beide Kantenrichtungen $s \to t$ (d.h. $v \to u$ und $u \to v$) und jeden Zustand $x \in \mathcal{X}_t$:

$$m^{\text{neu}}_{v \to u}(x) = \sum_{y \in \mathcal{X}_v} \exp(\langle \boldsymbol{\beta}_{uv}, \phi_{uv}(x, y) \rangle) \prod_{w \in \mathcal{N}(v) \setminus \{u\}} m^{\text{alt}}_{w \to v}(y)$$

(3) Falls $\|m^{\text{neu}} - m^{\text{alt}}\| > \varepsilon$: $m^{\text{alt}} \leftarrow m^{\text{neu}}$ und GOTO (2)

Loopy Belief Propagation

- Hier: Einschränkung auf Graphen mit $\max_{C \in \mathcal{C}(G)} |C| = 2$
- Dann: Menge der Cliquen $\mathcal{C}(G)$ = Menge der Kanten E
- Der Gradient enthält dann $\mathbb{P}(\boldsymbol{X}_{\{v,u\}} = \boldsymbol{y})$ für alle $\{v, u\} \in E$ und alle $\boldsymbol{y} \in \mathcal{X}_{\{v,u\}}$

Algorithmus:

(1) Lege zwei Arrays m^{neu} und m^{alt} an (jeweils $\sum_{\{v,u\} \in E} (|\mathcal{X}|_v + |\mathcal{X}|_u)$ Einträge) und initialisiere mit 1

(2) Berechne für alle Kanten $\{v, u\}$ in E, beide Kantenrichtungen $s \to t$ (d.h. $v \to u$ und $u \to v$) und jeden Zustand $x \in \mathcal{X}_t$:

$$m^{\text{neu}}_{v \to u}(x) = \sum_{y \in \mathcal{X}_v} \exp(\langle \boldsymbol{\beta}_{uv}, \phi_{uv}(x, y) \rangle) \prod_{w \in \mathcal{N}(v) \setminus \{u\}} m^{\text{alt}}_{w \to v}(y)$$

(3) Falls $\|m^{\text{neu}} - m^{\text{alt}}\| > \varepsilon$: $m^{\text{alt}} \leftarrow m^{\text{neu}}$ und GOTO (2)

Loopy Belief Propagation

- Hier: Einschränkung auf Graphen mit $\max_{C \in \mathcal{C}(G)} |C| = 2$
- Dann: Menge der Cliquen $\mathcal{C}(G)$ = Menge der Kanten E
- Der Gradient enthält dann $\mathbb{P}(\boldsymbol{X}_{\{v,u\}} = \boldsymbol{y})$ für alle $\{v, u\} \in E$ und alle $\boldsymbol{y} \in \mathcal{X}_{\{v,u\}}$

Algorithmus:

(1) Lege zwei Arrays m^{neu} und m^{alt} an (jeweils $\sum_{\{v,u\} \in E} (|\mathcal{X}|_v + |\mathcal{X}|_u)$ Einträge) und initialisiere mit 1

(2) Berechne für alle Kanten $\{v, u\}$ in E, beide Kantenrichtungen $s \to t$ (d.h. $v \to u$ und $u \to v$) und jeden Zustand $x \in \mathcal{X}_t$:

$$m^{\text{neu}}_{v \to u}(x) = \sum_{y \in \mathcal{X}_v} \exp(\langle \boldsymbol{\beta}_{uv}, \phi_{uv}(x, y) \rangle) \prod_{w \in \mathcal{N}(v) \setminus \{u\}} m^{\text{alt}}_{w \to v}(y)$$

(3) Falls $\|m^{\text{neu}} - m^{\text{alt}}\| > \varepsilon$: $m^{\text{alt}} \leftarrow m^{\text{neu}}$ und GOTO (2)

Loopy Belief Propagation

- Hier: Einschränkung auf Graphen mit $\max_{C \in \mathcal{C}(G)} |C| = 2$
- Dann: Menge der Cliquen $\mathcal{C}(G)$ = Menge der Kanten E
- Der Gradient enthält dann $\mathbb{P}(\boldsymbol{X}_{\{v,u\}} = \boldsymbol{y})$ für alle $\{v, u\} \in E$ und alle $\boldsymbol{y} \in \mathcal{X}_{\{v,u\}}$

Algorithmus:

(1) Lege zwei Arrays m^{neu} und m^{alt} an (jeweils $\sum_{\{v,u\} \in E} (|\mathcal{X}|_v + |\mathcal{X}|_u)$ Einträge) und initialisiere mit 1

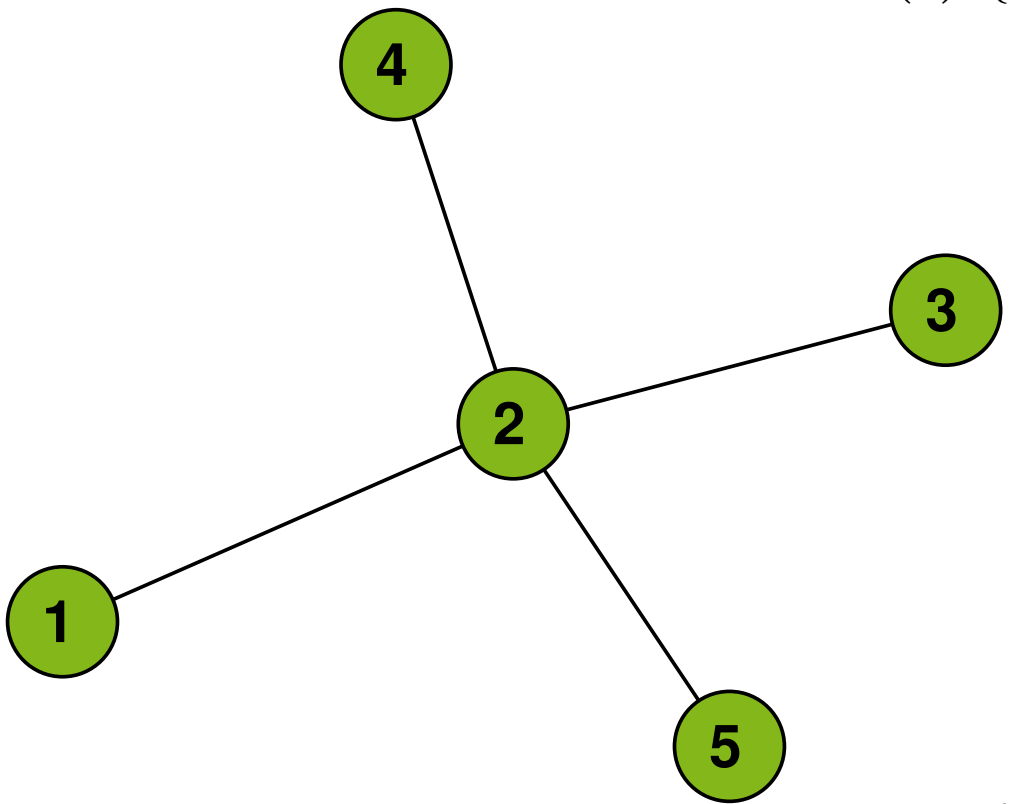
(2) Berechne für alle Kanten $\{v, u\}$ in E, beide Kantenrichtungen $s \to t$ (d.h. $v \to u$ und $u \to v$) und jeden Zustand $x \in \mathcal{X}_t$:

$$m^{\text{neu}}_{v \to u}(x) = \sum_{y \in \mathcal{X}_v} \exp(\langle \boldsymbol{\beta}_{uv}, \phi_{uv}(x, y) \rangle) \prod_{w \in \mathcal{N}(v) \setminus \{u\}} m^{\text{alt}}_{w \to v}(y)$$

(3) Falls $\|m^{\text{neu}} - m^{\text{alt}}\| > \varepsilon$: $m^{\text{alt}} \leftarrow m^{\text{neu}}$ und GOTO (2)

Loopy Belief Propagation: Visualisierung (1. Iteration)

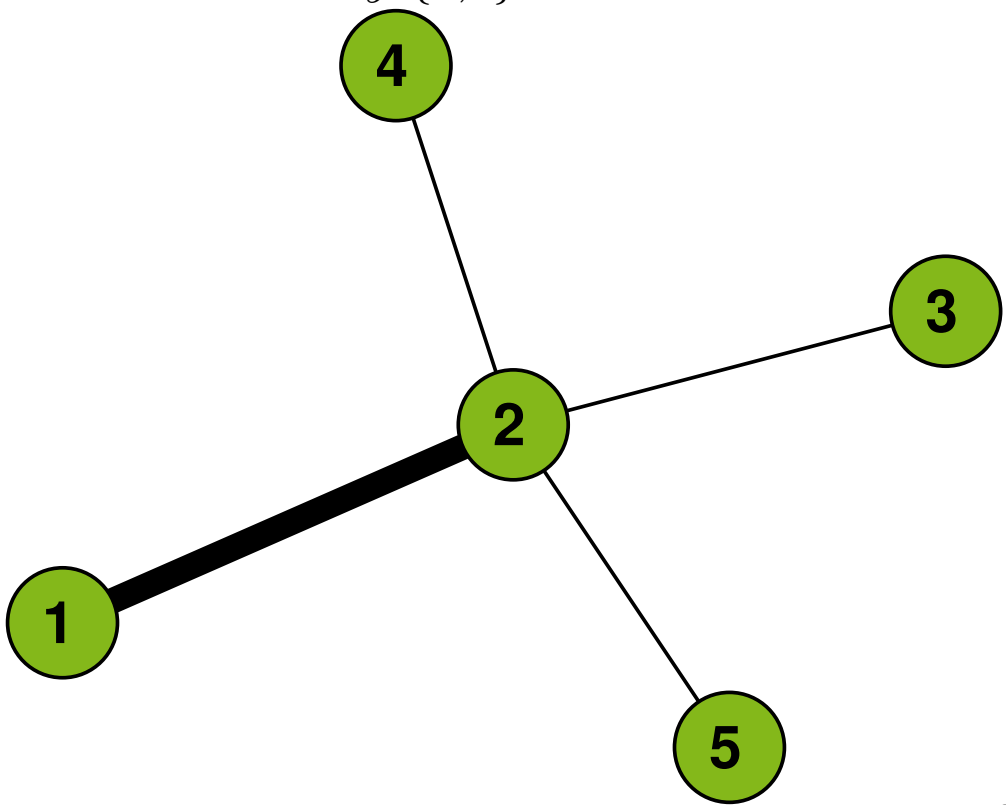
$$m_{v\to u}^{\text{neu}}(x) = \sum_{y\in\mathcal{X}_v} \exp(\langle \boldsymbol{\beta}_{uv}, \phi_{uv}(x,y)\rangle) \prod_{w\in\mathcal{N}(v)\setminus\{u\}} m_{w\to v}^{\text{alt}}(y)$$



$$\mathcal{X}_v = \{0,1\}, v \in \{1,2,\ldots,5\}$$

Loopy Belief Propagation: Visualisierung (1. Iteration)

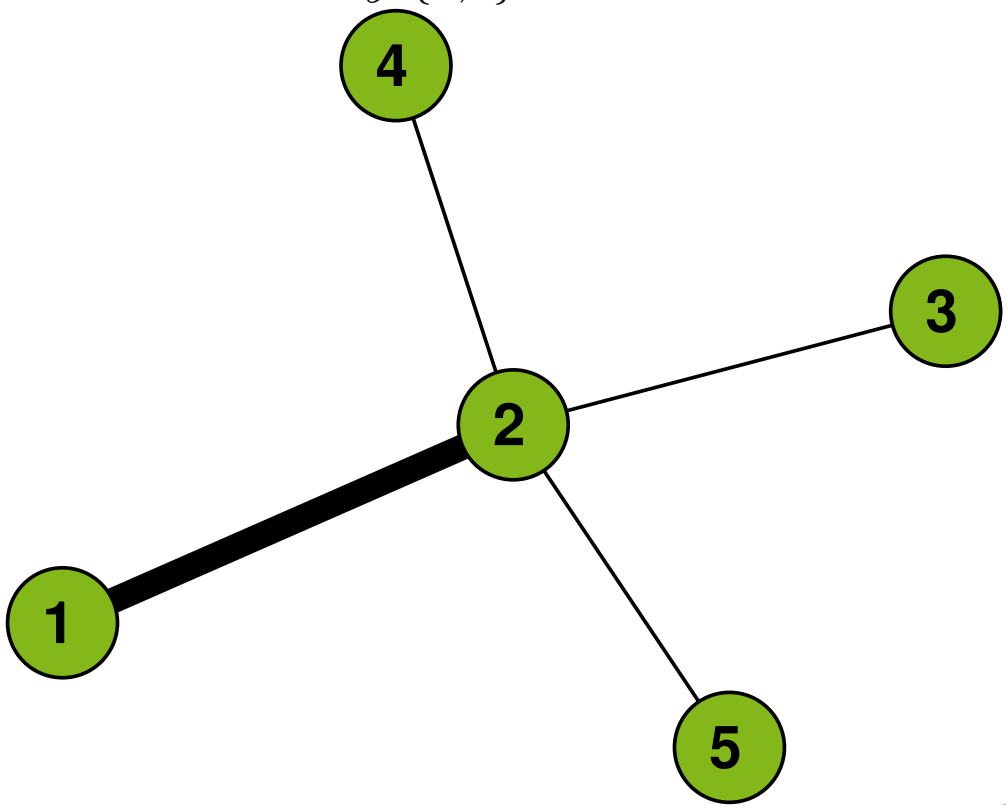
$$m_{1\to 2}^{\text{neu}}(0) = \sum_{y\in\{0,1\}} \exp(\langle \boldsymbol{\beta}_{12}, \phi_{12}(y,0)\rangle)$$



$$\mathcal{X}_v = \{0,1\}, v \in \{1,2,\ldots,5\}$$

Loopy Belief Propagation: Visualisierung (1. Iteration)

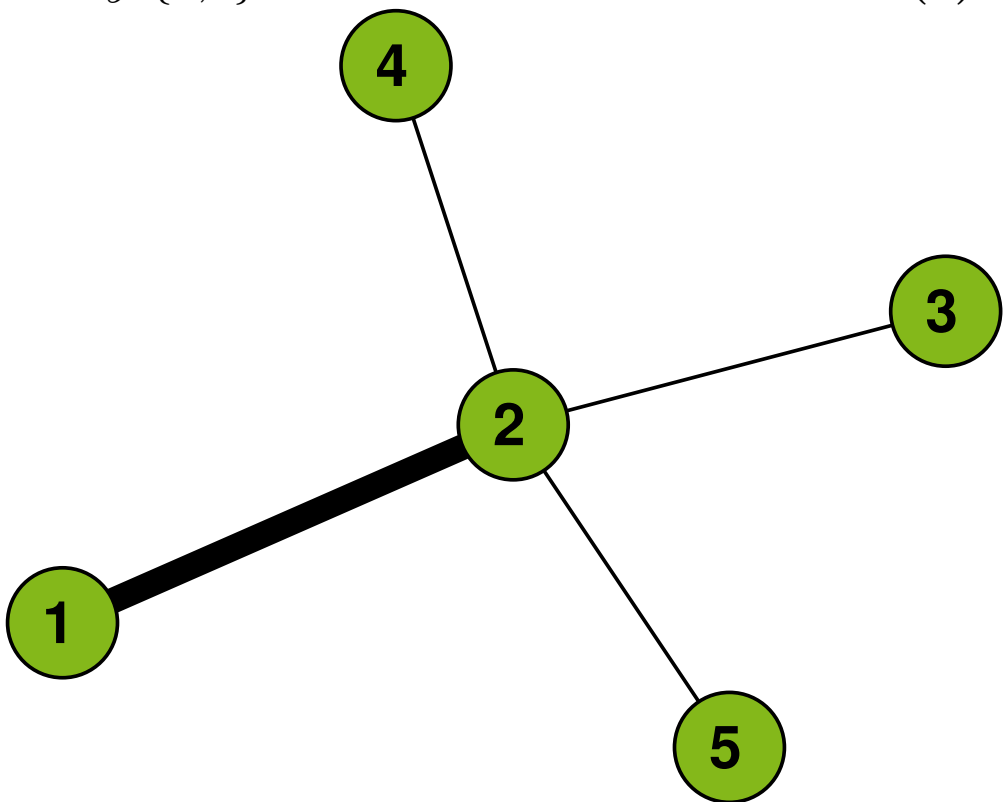
$$m_{1\to 2}^{\text{neu}}(1) = \sum_{y\in\{0,1\}} \exp(\langle \boldsymbol{\beta}_{12}, \phi_{12}(y,1)\rangle)$$



$$\mathcal{X}_v = \{0,1\}, v \in \{1,2,\ldots,5\}$$

Loopy Belief Propagation: Visualisierung (1. Iteration)

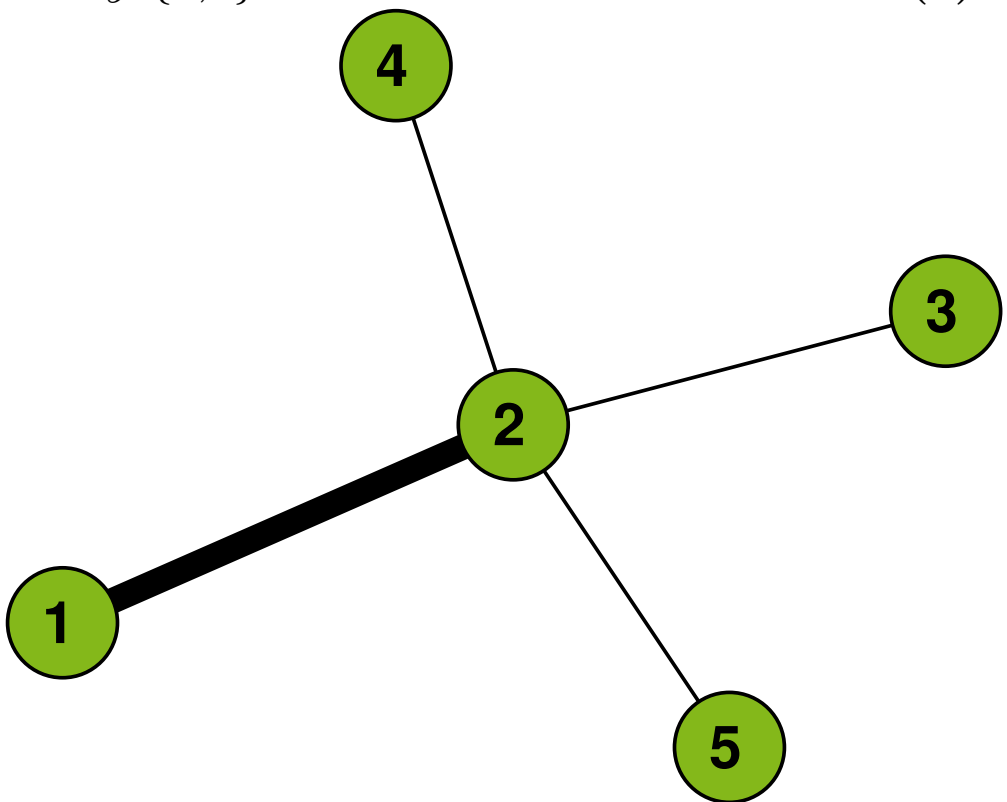
$$m_{2\to 1}^{\text{neu}}(0) = \sum_{y\in\{0,1\}} \exp(\langle \beta_{12}, \phi_{12}(0,y)\rangle) \prod_{w\in\mathcal{N}(v)\setminus\{1\}} m_{w\to 2}^{\text{alt}}(y)$$



$$\mathcal{X}_v = \{0,1\}, v \in \{1,2,\ldots,5\}$$

Loopy Belief Propagation: Visualisierung (1. Iteration)

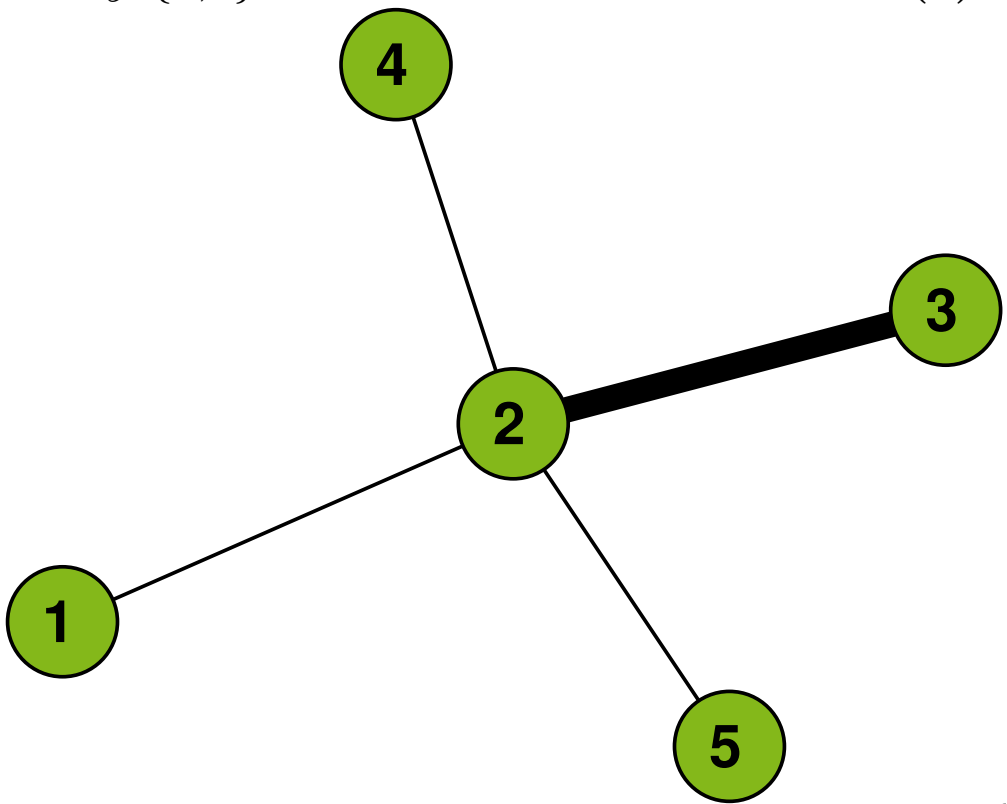
$$m_{2\to 1}^{\text{neu}}(1) = \sum_{y\in\{0,1\}} \exp(\langle \beta_{12}, \phi_{12}(1,y)\rangle) \prod_{w\in\mathcal{N}(v)\setminus\{1\}} m_{w\to 2}^{\text{alt}}(y)$$



$$\mathcal{X}_v = \{0,1\}, v \in \{1,2,\ldots,5\}$$

Loopy Belief Propagation: Visualisierung (1. Iteration)

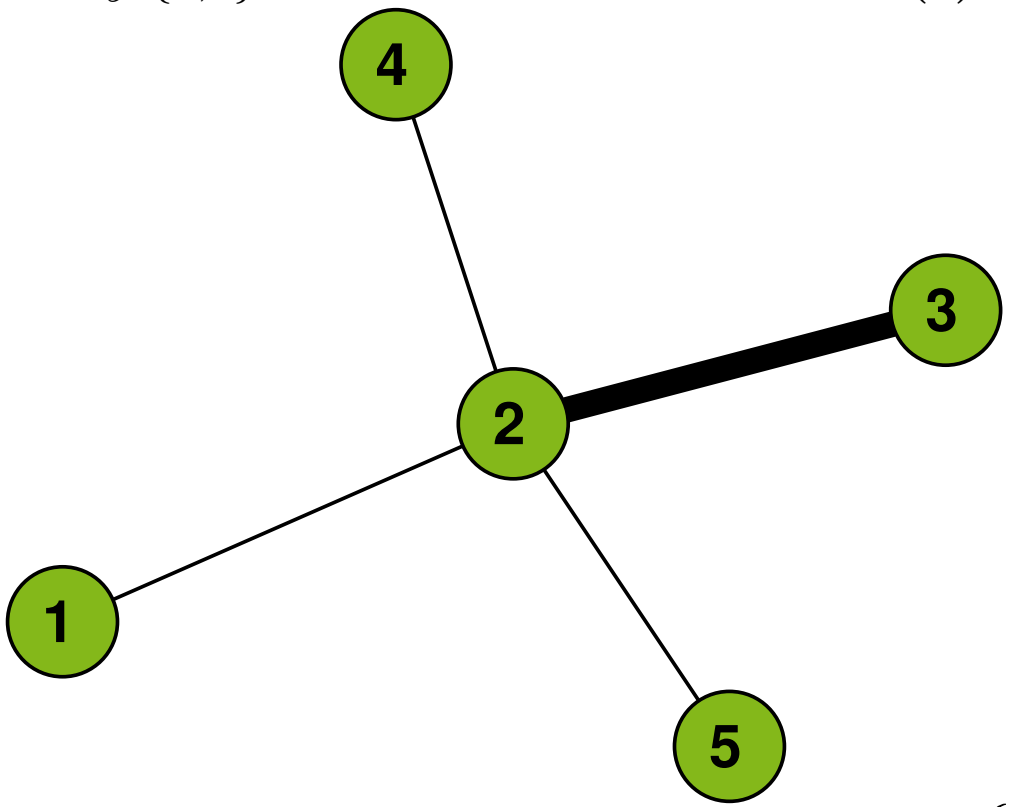
$$m_{2\to 3}^{\text{neu}}(0) = \sum_{y\in\{0,1\}} \exp(\langle \beta_{23}, \phi_{23}(y,0)\rangle) \prod_{w\in\mathcal{N}(v)\setminus\{3\}} m_{w\to 2}^{\text{alt}}(y)$$



$$\mathcal{X}_v = \{0,1\}, v \in \{1,2,\ldots,5\}$$

Loopy Belief Propagation: Visualisierung (1. Iteration)

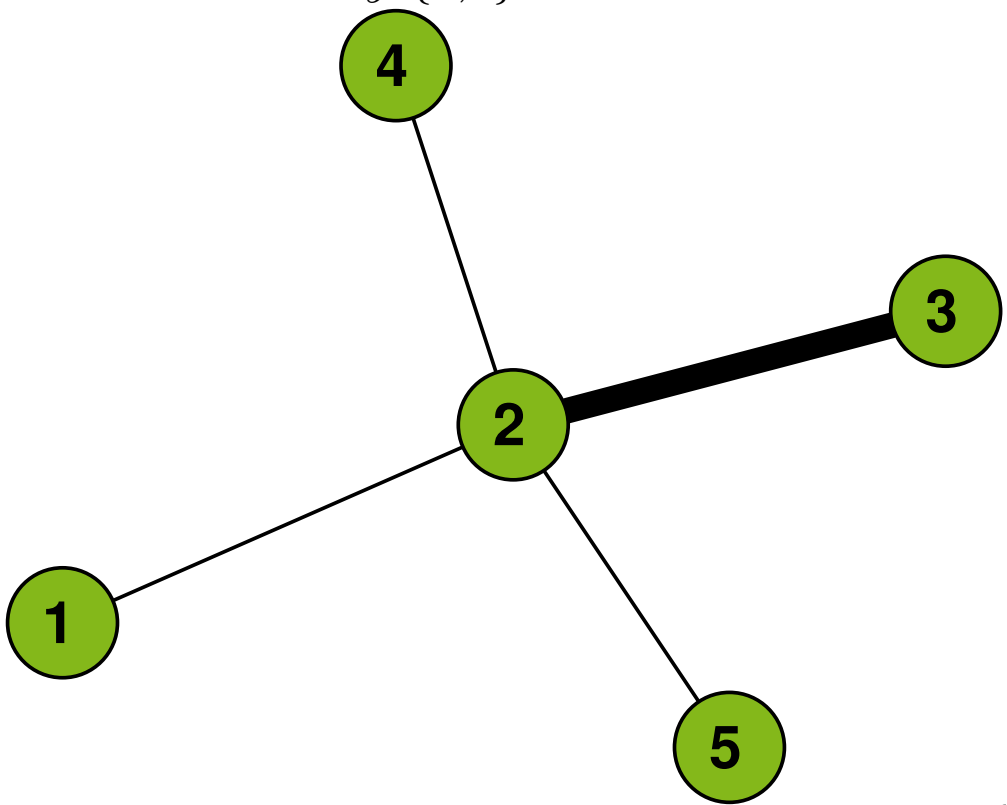
$$m_{2\to 3}^{\text{neu}}(1) = \sum_{y\in\{0,1\}} \exp(\langle \boldsymbol{\beta}_{23}, \phi_{23}(y,1)\rangle) \prod_{w\in\mathcal{N}(v)\setminus\{3\}} m_{w\to 2}^{\text{alt}}(y)$$



$$\mathcal{X}_v = \{0,1\}, v \in \{1,2,\ldots,5\}$$

Loopy Belief Propagation: Visualisierung (1. Iteration)

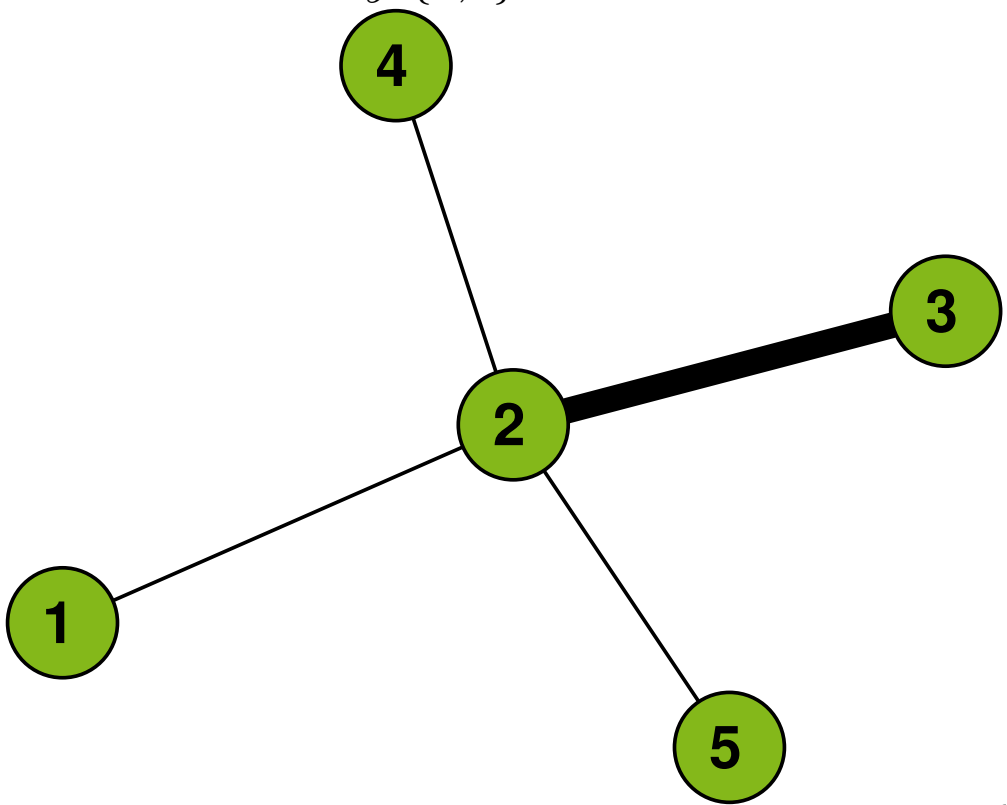
$$m_{3\to 2}^{\text{neu}}(0) = \sum_{y\in\{0,1\}} \exp(\langle \boldsymbol{\beta}_{23}, \phi_{23}(0,y)\rangle)$$



$$\mathcal{X}_v = \{0,1\}, v \in \{1,2,\ldots,5\}$$

Loopy Belief Propagation: Visualisierung (1. Iteration)

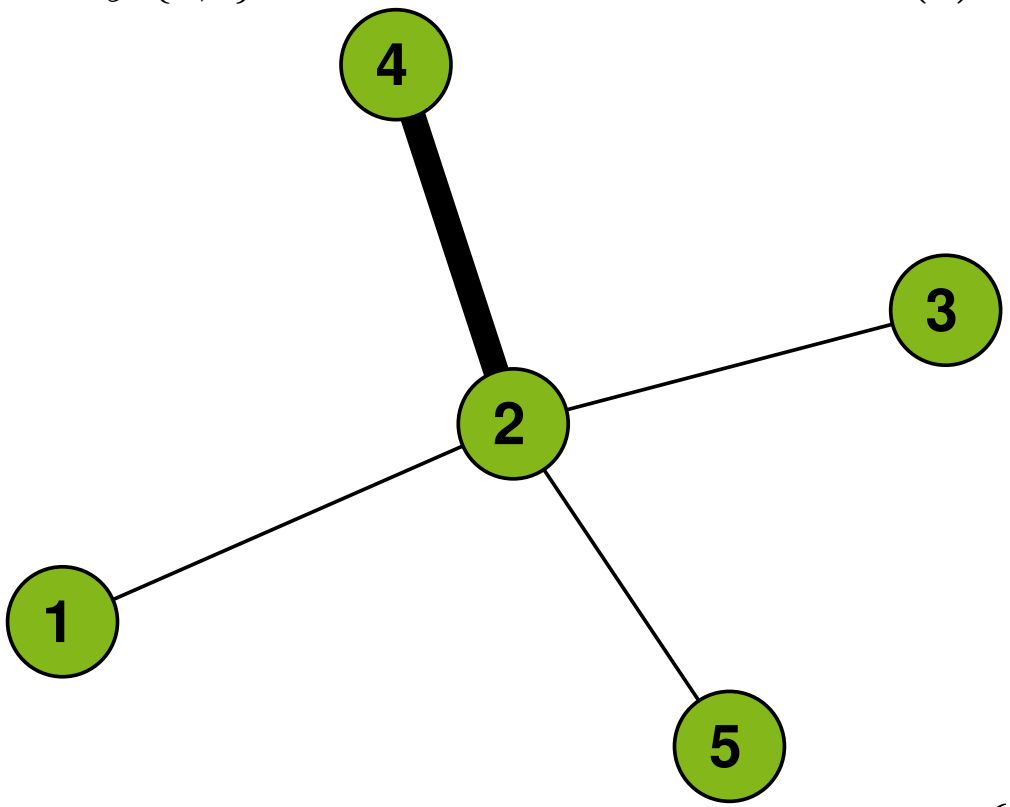
$$m_{3\to 2}^{\text{neu}}(1) = \sum_{y\in\{0,1\}} \exp(\langle \boldsymbol{\beta}_{23}, \phi_{23}(1,y)\rangle)$$



$$\mathcal{X}_v = \{0,1\}, v \in \{1,2,\ldots,5\}$$

Loopy Belief Propagation: Visualisierung (1. Iteration)

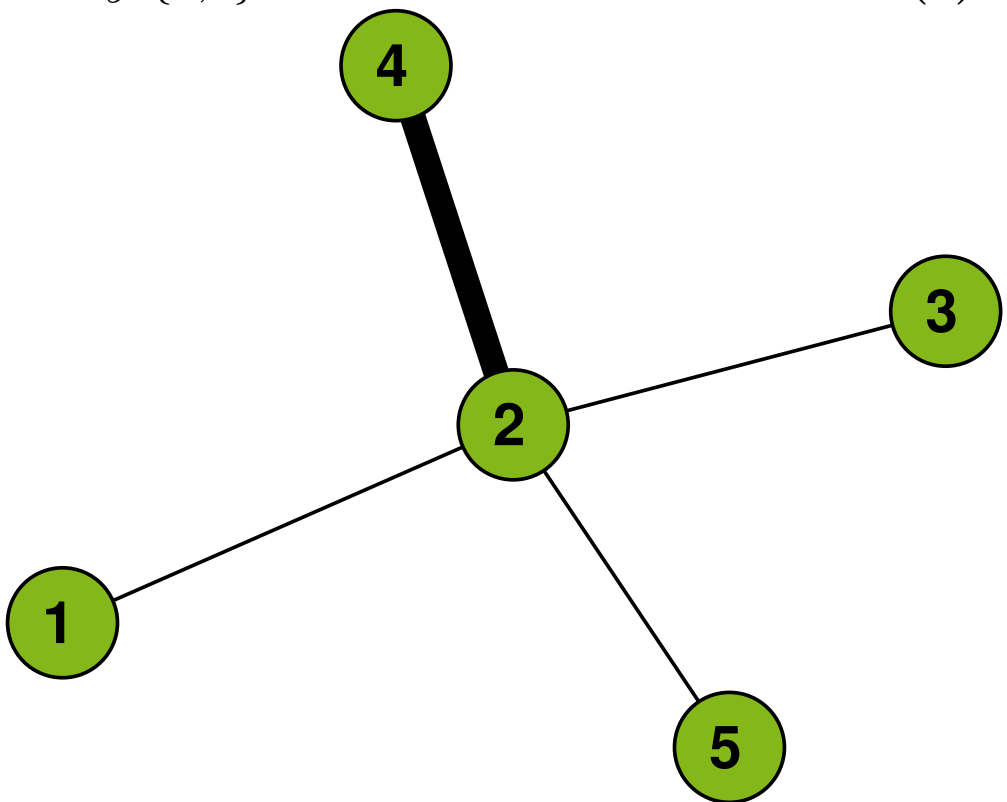
$$m_{2\to 4}^{\text{neu}}(0) = \sum_{y\in\{0,1\}} \exp(\langle \beta_{24}, \phi_{24}(y,0)\rangle) \prod_{w\in\mathcal{N}(v)\setminus\{4\}} m_{w\to 2}^{\text{alt}}(y)$$



$$\mathcal{X}_v = \{0,1\}, v \in \{1,2,\ldots,5\}$$

Loopy Belief Propagation: Visualisierung (1. Iteration)

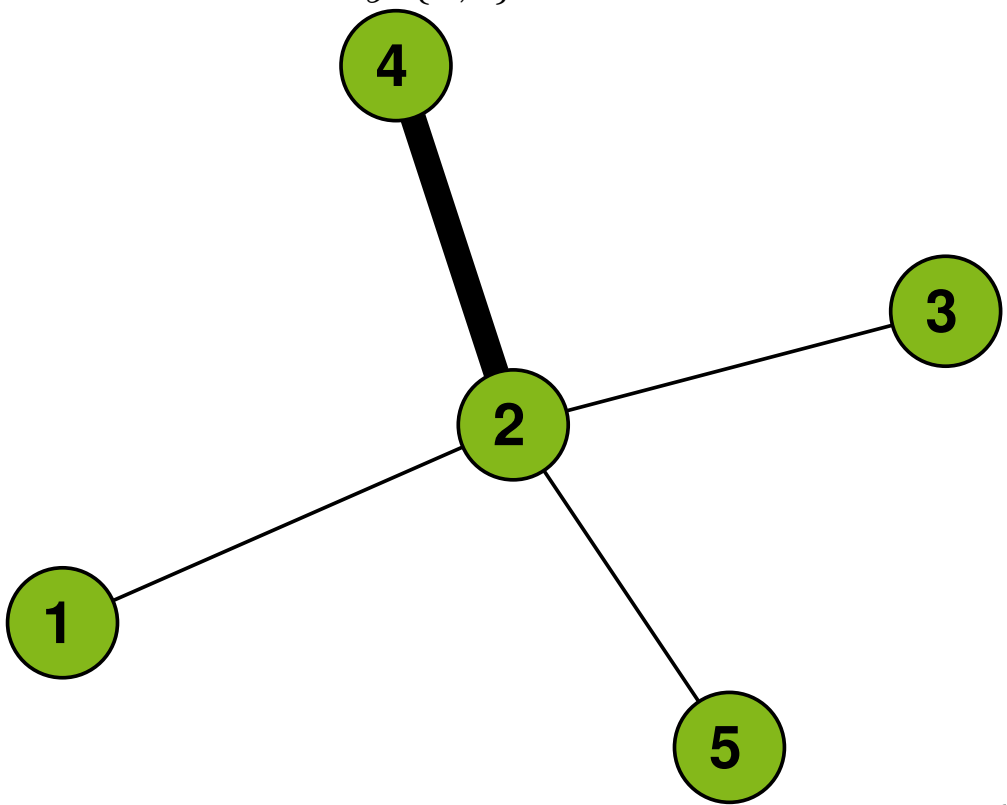
$$m_{2\to 4}^{\text{neu}}(1) = \sum_{y\in\{0,1\}} \exp(\langle \beta_{24}, \phi_{24}(y,1)\rangle) \prod_{w\in\mathcal{N}(v)\setminus\{4\}} m_{w\to 2}^{\text{alt}}(y)$$



$$\mathcal{X}_v = \{0,1\}, v \in \{1,2,\ldots,5\}$$

Loopy Belief Propagation: Visualisierung (1. Iteration)

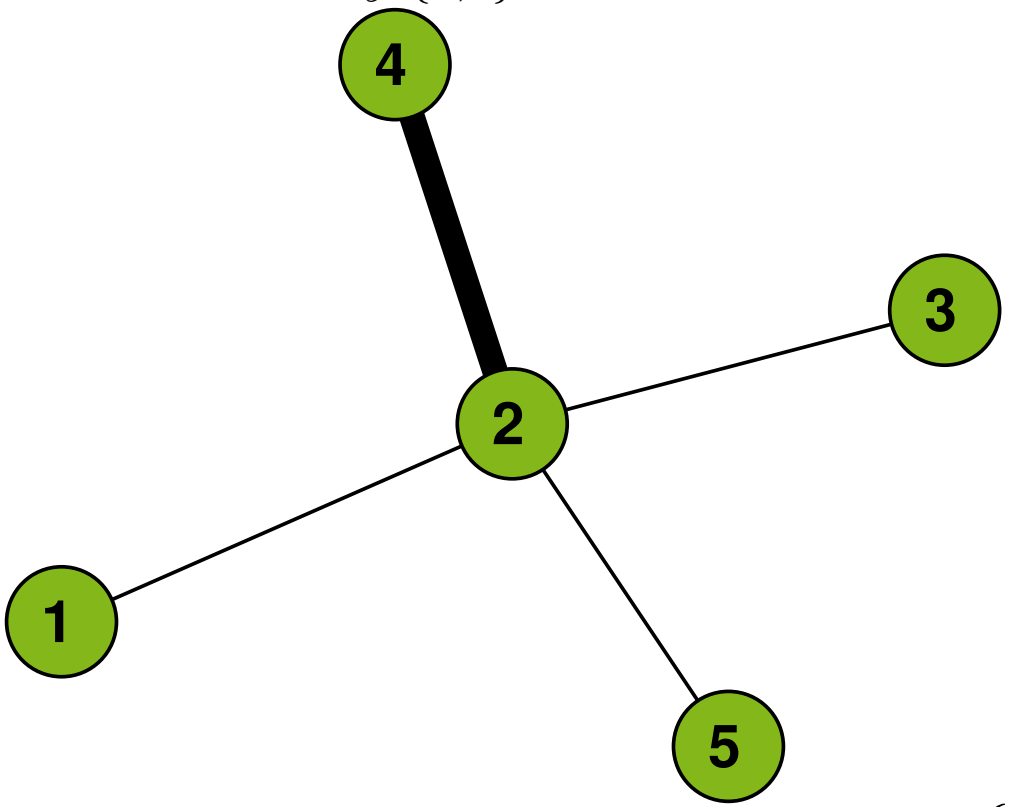
$$m_{4\to 2}^{\text{neu}}(0) = \sum_{y\in\{0,1\}} \exp(\langle \boldsymbol{\beta}_{24}, \phi_{24}(0,y)\rangle)$$



$$\mathcal{X}_v = \{0,1\}, v \in \{1,2,\ldots,5\}$$

Loopy Belief Propagation: Visualisierung (1. Iteration)

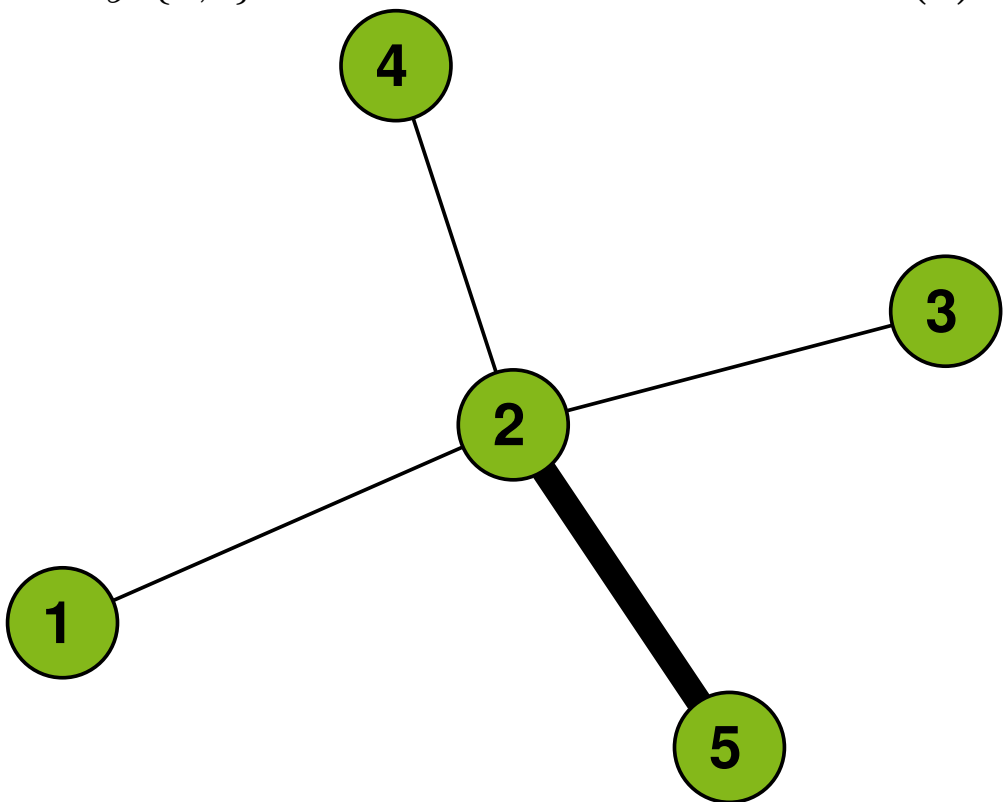
$$m_{4\to 2}^{\text{neu}}(1) = \sum_{y\in\{0,1\}} \exp(\langle \boldsymbol{\beta}_{24}, \phi_{24}(1,y)\rangle)$$



$$\mathcal{X}_v = \{0,1\}, v \in \{1,2,\ldots,5\}$$

Loopy Belief Propagation: Visualisierung (1. Iteration)

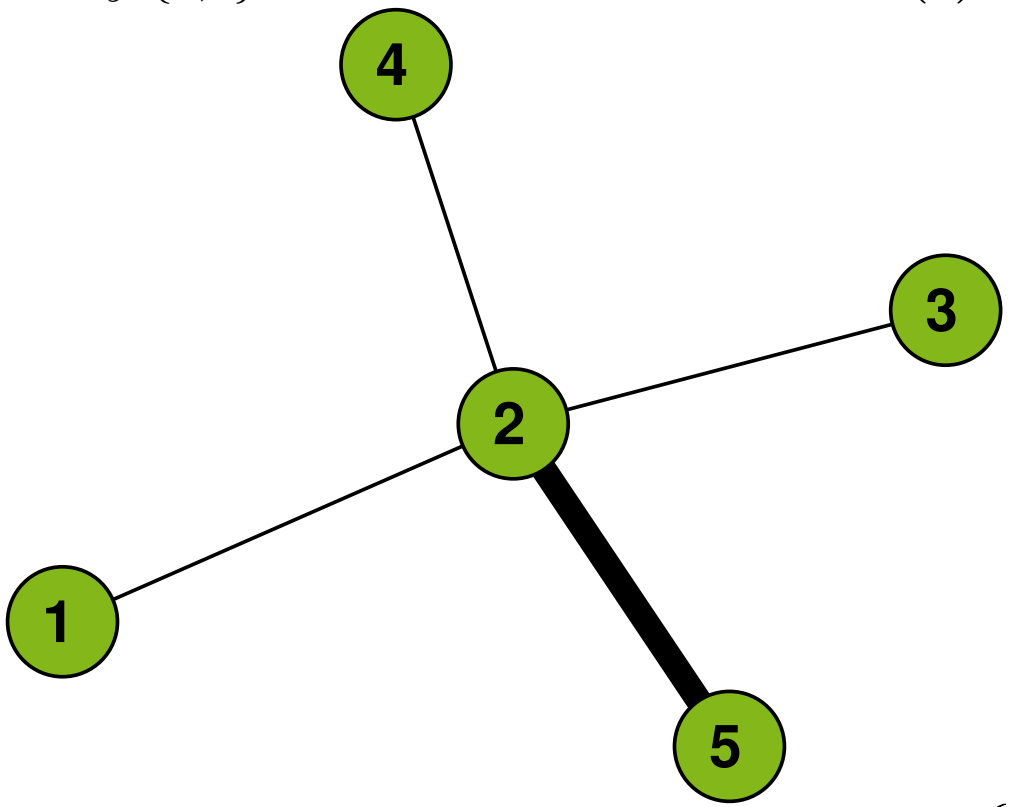
$$m_{2\to 5}^{\text{neu}}(0) = \sum_{y\in\{0,1\}} \exp(\langle \boldsymbol{\beta}_{25}, \phi_{25}(y,0)\rangle) \prod_{w\in\mathcal{N}(v)\setminus\{5\}} m_{w\to 2}^{\text{alt}}(y)$$



$$\mathcal{X}_v = \{0,1\}, v \in \{1,2,\ldots,5\}$$

Loopy Belief Propagation: Visualisierung (1. Iteration)

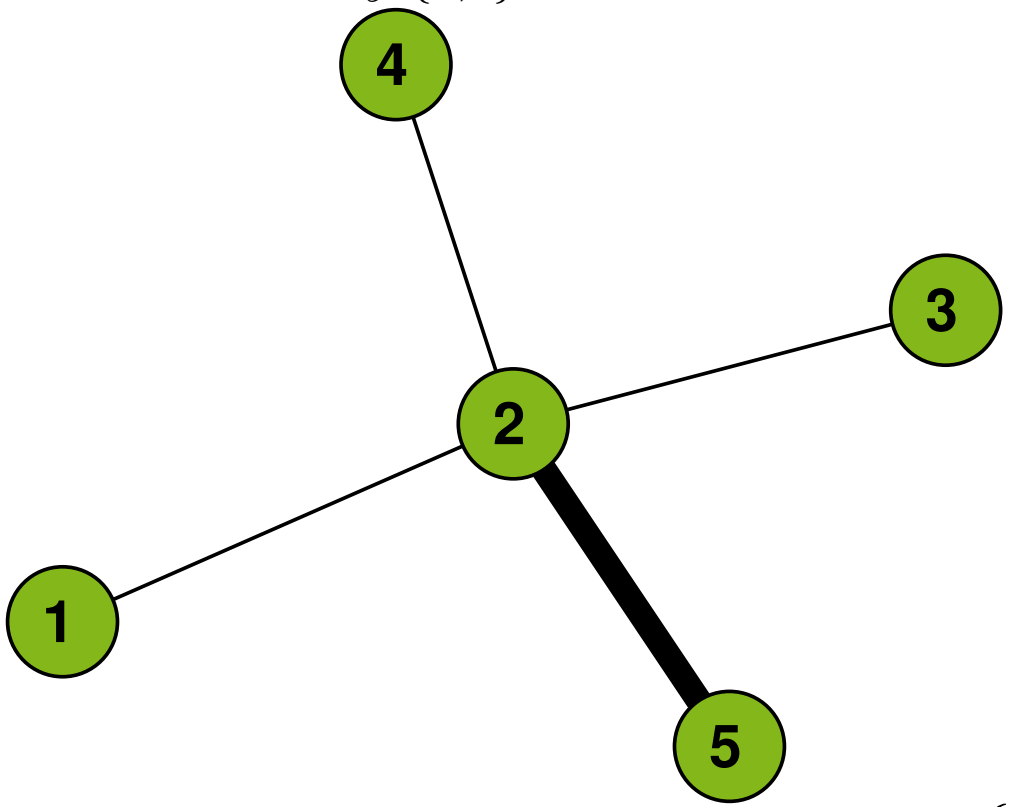
$$m_{2\to 5}^{\text{neu}}(1) = \sum_{y\in\{0,1\}} \exp(\langle \boldsymbol{\beta}_{25}, \phi_{25}(y,1)\rangle) \prod_{w\in\mathcal{N}(v)\setminus\{5\}} m_{w\to 2}^{\text{alt}}(y)$$



$$\mathcal{X}_v = \{0,1\}, v \in \{1,2,\ldots,5\}$$

Loopy Belief Propagation: Visualisierung (1. Iteration)

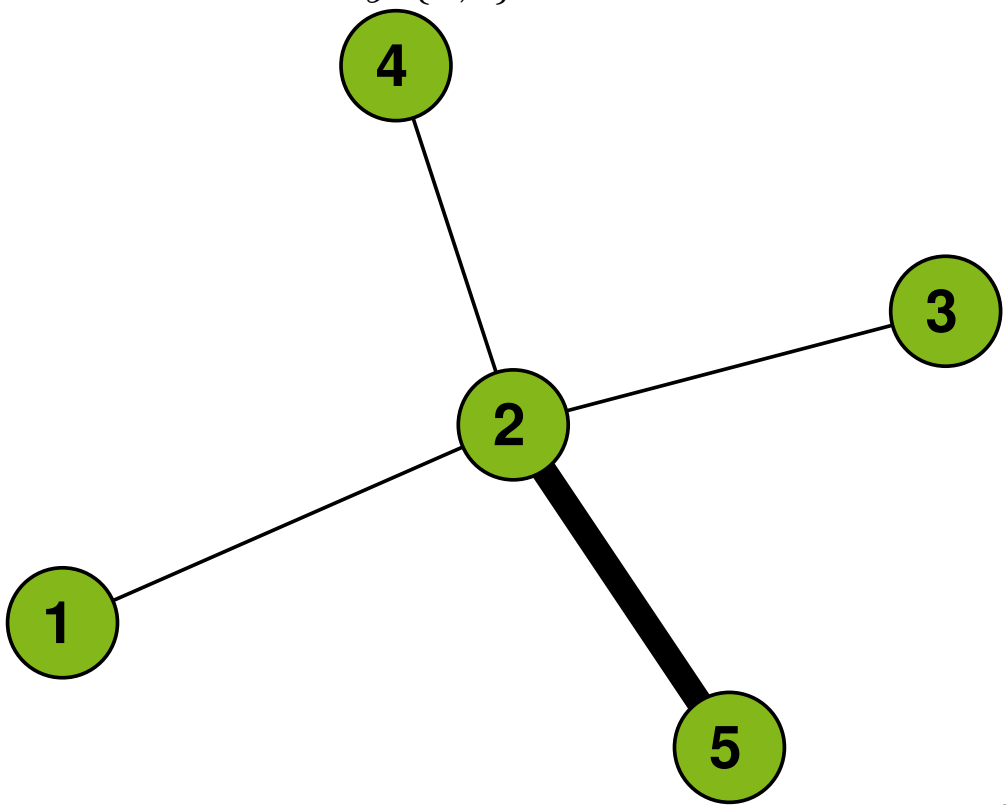
$$m_{5\to 2}^{\text{neu}}(0) = \sum_{y\in\{0,1\}} \exp(\langle \boldsymbol{\beta}_{25}, \phi_{25}(0,y)\rangle)$$



$$\mathcal{X}_v = \{0,1\}, v \in \{1,2,\ldots,5\}$$

Loopy Belief Propagation: Visualisierung (1. Iteration)

$$m_{5\to 2}^{\text{neu}}(1) = \sum_{y\in\{0,1\}} \exp(\langle \boldsymbol{\beta}_{25}, \phi_{25}(1,y)\rangle)$$



$$\mathcal{X}_v = \{0,1\}, v \in \{1,2,\ldots,5\}$$

Loopy Belief Propagation: “Konvergenz”

Nachdem m^{neu} basierend auf m^{neu} berechnet wurde, überprüfen wir die Konvergenz mittels

$$\|m^{\text{neu}} - m^{\text{alt}}\| > \varepsilon$$

Die obige Bedingung kann für jedes $\varepsilon \geq 0$ erfüllt werden, falls

- der Graph G ein Baum ist

Andernfalls muss ε “groß genug” (anwendungsspezifisch) gewählt werden.

Loopy Belief Propagation: “Konvergenz”

Nachdem m^{neu} basierend auf m^{neu} berechnet wurde, überprüfen wir die Konvergenz mittels

$$\|m^{\text{neu}} - m^{\text{alt}}\| > \varepsilon$$

Die obige Bedingung kann für jedes $\varepsilon \geq 0$ erfüllt werden, falls

- der Graph G ein Baum ist

Andernfalls muss ε “groß genug” (anwendungsspezifisch) gewählt werden.

Strukturlernen

"Woher kommt der Graph?"

- **Gegeben**: Menge an Varaiblen V, Datensatz $\mathcal{D} \rightarrow \tilde{\mathbb{E}}[\phi(\boldsymbol{X})] = \frac{1}{|\mathcal{D}|} \sum_{\boldsymbol{x} \in \mathcal{D}} \phi(\boldsymbol{x})$
- **Idee**: Ausnutzen bedingter Unabhängigkeiten zwischen den Variablen

Erinnerung:

$$\boldsymbol{X}_v \perp\!\!\!\perp \boldsymbol{X}_{V \setminus (\mathcal{N}(v) \cup \{v\})} \mid \boldsymbol{X}_{\mathcal{N}(v)}, \forall v \in V$$

$$\boldsymbol{X}_U \perp\!\!\!\perp \boldsymbol{X}_W \mid \boldsymbol{X}_S$$

$\forall U, W \subset V$, falls jeder Pfad von U nach W durch S führt.

Strukturlernen

"Woher kommt der Graph?"

- **Gegeben**: Menge an Varaiblen V, Datensatz $\mathcal{D} \rightarrow \tilde{\mathbb{E}}[\phi(\boldsymbol{X})] = \frac{1}{|\mathcal{D}|} \sum_{\boldsymbol{x} \in \mathcal{D}} \phi(\boldsymbol{x})$
- **Idee**: Ausnutzen bedingter Unabhängigkeiten zwischen den Variablen

Erinnerung:

$$\boldsymbol{X}_v \perp\!\!\!\perp \boldsymbol{X}_{V \setminus (\mathcal{N}(v) \cup \{v\})} \mid \boldsymbol{X}_{\mathcal{N}(v)}, \forall v \in V$$

$$\boldsymbol{X}_U \perp\!\!\!\perp \boldsymbol{X}_W \mid \boldsymbol{X}_S$$

$\forall U, W \subset V$, falls jeder Pfad von U nach W durch S führt.

Strukturlernen

"Woher kommt der Graph?"

- **Gegeben**: Menge an Varaiblen V, Datensatz $\mathcal{D} \rightarrow \tilde{\mathbb{E}}[\phi(\boldsymbol{X})] = \frac{1}{|\mathcal{D}|} \sum_{\boldsymbol{x} \in \mathcal{D}} \phi(\boldsymbol{x})$
- **Idee**: Ausnutzen bedingter Unabhängigkeiten zwischen den Variablen

Erinnerung:

$$\boldsymbol{X}_v \perp\!\!\!\perp \boldsymbol{X}_{V \setminus (\mathcal{N}(v) \cup \{v\})} \mid \boldsymbol{X}_{\mathcal{N}(v)}, \forall v \in V$$

$$\boldsymbol{X}_U \perp\!\!\!\perp \boldsymbol{X}_W \mid \boldsymbol{X}_S$$

$\forall U, W \subset V$, falls jeder Pfad von U nach W durch S führt.

Option1: Chow-Liu Bäume

Idee: Finde besten **Baum** T, so dass Abstand (Kullback-Leiber Divergenz) zwischen echter Funktion $\mathbb{P}^*$ und $\mathbb{P}_T$ minimal ist.

$$\min_T \mathrm{KL}(\mathbb{P}^*, \mathbb{P}_T) = \min_T \sum_{\boldsymbol{x} \in \mathcal{X}} \mathbb{P}^*(\boldsymbol{x}) \log \frac{\mathbb{P}^*(\boldsymbol{x})}{\mathbb{P}_T(\boldsymbol{x})}$$

Kann umgeform werden zu

$$\mathrm{KL}(\mathbb{P}^*, \mathbb{P}_T) = -\mathcal{H}(\mathbb{P}^*) + \sum_{v \in V} \mathcal{H}(\mathbb{P}^*_v) - \underbrace{\sum_{vu \in E(T)} I(\boldsymbol{X}_v, \boldsymbol{X}_u)}_{\text{Maximaler Spannbaum!}}$$

Option1: Chow-Liu Bäume

Idee: Finde besten **Baum** T, so dass Abstand (Kullback-Leiber Divergenz) zwischen echter Funktion $\mathbb{P}^*$ und $\mathbb{P}_T$ minimal ist.

$$\min_T \mathsf{KL}(\mathbb{P}^*, \mathbb{P}_T) = \min_T \sum_{\boldsymbol{x} \in \mathcal{X}} \mathbb{P}^*(\boldsymbol{x}) \log \frac{\mathbb{P}^*(\boldsymbol{x})}{\mathbb{P}_T(\boldsymbol{x})}$$

Kann umgeform werden zu

$$\mathsf{KL}(\mathbb{P}^*, \mathbb{P}_T) = -\mathcal{H}(\mathbb{P}^*) + \sum_{v \in V} \mathcal{H}(\mathbb{P}_v^*) - \underbrace{\sum_{vu \in E(T)} I(\boldsymbol{X}_v, \boldsymbol{X}_u)}_{\text{Maximaler Spannbaum!}}$$

Option1: Chow-Liu Bäume

Idee: Finde besten **Baum** T, so dass Abstand (Kullback-Leiber Divergenz) zwischen echter Funktion $\mathbb{P}^*$ und $\mathbb{P}_T$ minimal ist.

$$\min_T \mathsf{KL}(\mathbb{P}^*, \mathbb{P}_T) = \min_T \sum_{\boldsymbol{x} \in \mathcal{X}} \mathbb{P}^*(\boldsymbol{x}) \log \frac{\mathbb{P}^*(\boldsymbol{x})}{\mathbb{P}_T(\boldsymbol{x})}$$

Kann umgeform werden zu

$$\mathsf{KL}(\mathbb{P}^*, \mathbb{P}_T) = -\mathcal{H}(\mathbb{P}^*) + \sum_{v \in V} \mathcal{H}(\mathbb{P}_v^*) - \underbrace{\sum_{vu \in E(T)} I(\boldsymbol{X}_v, \boldsymbol{X}_u)}_{\text{Maximaler Spannbaum!}}$$

Mutual Information

Erinnerung: Zwei Variablen $\boldsymbol{X}_v$ und $\boldsymbol{X}_u$ sind unabhängig, falls $\mathbb{P}(\boldsymbol{X}_v = x, \boldsymbol{X}_u = y) = \mathbb{P}(\boldsymbol{X}_v = x)\mathbb{P}(\boldsymbol{X}_u = y)$ für alle x, y. Das heißt falls

$$\frac{\mathbb{P}(\boldsymbol{X}_v = x, \boldsymbol{X}_u = y)}{\mathbb{P}(\boldsymbol{X}_v = x)\mathbb{P}(\boldsymbol{X}_u = y)} = 1, \forall x \in \mathcal{X}_v, y \in \mathcal{X}_y$$

Das gilt möglicherweise nicht für alle x, y, **aber**:
Allgemeines Maß für Unabhängigkeit durch Gewichtung jedes Paares x, y mittels $\mathbb{P}(\boldsymbol{X}_v = x, \boldsymbol{X}_u = y)$

$$\sum_{x \in \mathcal{X}_v} \sum_{y \in \mathcal{X}_u} \mathbb{P}(\boldsymbol{X}_v = x, \boldsymbol{X}_u = y) \frac{\mathbb{P}(\boldsymbol{X}_v = x, \boldsymbol{X}_u = y)}{\mathbb{P}(\boldsymbol{X}_v = x)\mathbb{P}(\boldsymbol{X}_u = y)} = I(\boldsymbol{X}_v, \boldsymbol{X}_u)$$

(I heißt “Mutual Information”)

Mutual Information

Erinnerung: Zwei Variablen $\boldsymbol{X}_v$ und $\boldsymbol{X}_u$ sind unabhängig, falls $\mathbb{P}(\boldsymbol{X}_v = x, \boldsymbol{X}_u = y) = \mathbb{P}(\boldsymbol{X}_v = x)\mathbb{P}(\boldsymbol{X}_u = y)$ für alle x, y. Das heißt falls

$$\frac{\mathbb{P}(\boldsymbol{X}_v = x, \boldsymbol{X}_u = y)}{\mathbb{P}(\boldsymbol{X}_v = x)\mathbb{P}(\boldsymbol{X}_u = y)} = 1, \forall x \in \mathcal{X}_v, y \in \mathcal{X}_y$$

Das gilt möglicherweise nicht für alle x, y, **aber**:

Allgemeines Maß für Unabhängigkeit durch Gewichtung jedes Paares x, y mittels $\mathbb{P}(\boldsymbol{X}_v = x, \boldsymbol{X}_u = y)$

$$\sum_{x \in \mathcal{X}_v} \sum_{y \in \mathcal{X}_u} \mathbb{P}(\boldsymbol{X}_v = x, \boldsymbol{X}_u = y) \frac{\mathbb{P}(\boldsymbol{X}_v = x, \boldsymbol{X}_u = y)}{\mathbb{P}(\boldsymbol{X}_v = x)\mathbb{P}(\boldsymbol{X}_u = y)} = I(\boldsymbol{X}_v, \boldsymbol{X}_u)$$

(I heißt “Mutual Information”)

Mutual Information

Erinnerung: Zwei Variablen $\boldsymbol{X}_v$ und $\boldsymbol{X}_u$ sind unabhängig, falls $\mathbb{P}(\boldsymbol{X}_v = x, \boldsymbol{X}_u = y) = \mathbb{P}(\boldsymbol{X}_v = x)\mathbb{P}(\boldsymbol{X}_u = y)$ für alle x, y. Das heißt falls

$$\frac{\mathbb{P}(\boldsymbol{X}_v = x, \boldsymbol{X}_u = y)}{\mathbb{P}(\boldsymbol{X}_v = x)\mathbb{P}(\boldsymbol{X}_u = y)} = 1, \forall x \in \mathcal{X}_v, y \in \mathcal{X}_y$$

Das gilt möglicherweise nicht für alle x, y, **aber**:
Allgemeines Maß für Unabhängigkeit durch Gewichtung jedes Paares x, y mittels $\mathbb{P}(\boldsymbol{X}_v = x, \boldsymbol{X}_u = y)$

$$\sum_{x \in \mathcal{X}_v} \sum_{y \in \mathcal{X}_u} \mathbb{P}(\boldsymbol{X}_v = x, \boldsymbol{X}_u = y) \frac{\mathbb{P}(\boldsymbol{X}_v = x, \boldsymbol{X}_u = y)}{\mathbb{P}(\boldsymbol{X}_v = x)\mathbb{P}(\boldsymbol{X}_u = y)} = I(\boldsymbol{X}_v, \boldsymbol{X}_u)$$

(I heißt “Mutual Information”)

Mutual Information

Erinnerung: Zwei Variablen $\boldsymbol{X}_v$ und $\boldsymbol{X}_u$ sind unabhängig, falls $\mathbb{P}(\boldsymbol{X}_v = x, \boldsymbol{X}_u = y) = \mathbb{P}(\boldsymbol{X}_v = x)\mathbb{P}(\boldsymbol{X}_u = y)$ für alle x, y. Das heißt falls

$$\frac{\mathbb{P}(\boldsymbol{X}_v = x, \boldsymbol{X}_u = y)}{\mathbb{P}(\boldsymbol{X}_v = x)\mathbb{P}(\boldsymbol{X}_u = y)} = 1, \forall x \in \mathcal{X}_v, y \in \mathcal{X}_y$$

Das gilt möglicherweise nicht für alle x, y, **aber**:
Allgemeines Maß für Unabhängigkeit durch Gewichtung jedes Paares x, y mittels $\mathbb{P}(\boldsymbol{X}_v = x, \boldsymbol{X}_u = y)$

$$\sum_{x \in \mathcal{X}_v} \sum_{y \in \mathcal{X}_u} \mathbb{P}(\boldsymbol{X}_v = x, \boldsymbol{X}_u = y) \frac{\mathbb{P}(\boldsymbol{X}_v = x, \boldsymbol{X}_u = y)}{\mathbb{P}(\boldsymbol{X}_v = x)\mathbb{P}(\boldsymbol{X}_u = y)} = I(\boldsymbol{X}_v, \boldsymbol{X}_u)$$

(I heißt “Mutual Information”)

Option1: Chow-Liu Bäume

Idee: Finde besten **Baum** T, so dass Abstand (Kullback-Leiber Divergenz) zwischen echter Funktion $\mathbb{P}^*$ und $\mathbb{P}^T$ minimal ist.

$$\min_T \mathsf{KL}(\mathbb{P}^*, \mathbb{P}_T) = \max_T \sum_{vu \in E(T)} I(\boldsymbol{X}_v, \boldsymbol{X}_u)$$

Berechnung beispielsweise mittels Algorithmus von Kruskal (Maximaler Spannbaum)

Option2: Graphen mittels Regularisierung

Baobachtung: Falls Parametervektor $\boldsymbol{\beta}_{\{v,u\}}$ einer Kante $= \mathbf{0}$, so hat diese Kante keinen Einfluss auf $\mathbb{P}(\boldsymbol{X} = \boldsymbol{x})$!

$$\mathbb{P}(\boldsymbol{X} = \boldsymbol{x}) = \frac{1}{Z(\boldsymbol{\beta})} \exp(\langle \boldsymbol{\beta}, \phi(\boldsymbol{x}) \rangle) = \frac{\prod_{C \in \mathcal{C}(G)} \exp(\langle \boldsymbol{\beta}_C, \phi_C(\boldsymbol{x}_C) \rangle)}{\sum_{\boldsymbol{x}' \in \mathcal{X}} \prod_{C \in \mathcal{C}(G)} \exp(\langle \boldsymbol{\beta}_C, \phi_C(\boldsymbol{x}'_C) \rangle)}$$

Falls alle Cliquenfaktoren in paarweise Funktionen zerfallen:

$$\mathbb{P}(\boldsymbol{X} = \boldsymbol{x}) = \frac{\prod_{\{v,u\} \in E} \exp(\langle \boldsymbol{\beta}_{\{v,u\}}, \phi_{\{v,u\}}(\boldsymbol{x}_{\{v,u\}}) \rangle)}{\sum_{\boldsymbol{x}' \in \mathcal{X}} \prod_{\{v,u\} \in E} \exp(\langle \boldsymbol{\beta}_{\{v,u\}}, \phi_{\{v,u\}}(\boldsymbol{x}'_{\{v,u\}}) \rangle)}$$

Falls $\boldsymbol{\beta}_{\{v,u\}} = \mathbf{0}$ für eine Kante $\{v, u\}$, so ist $\exp(\langle \boldsymbol{\beta}_{\{v,u\}}, \phi_{\{v,u\}}(\boldsymbol{x}_{\{v,u\}}) \rangle) = 1$.

Option2: Graphen mittels Regularisierung

Baobachtung: Falls Parametervektor $\boldsymbol{\beta}_{\{v,u\}}$ einer Kante $= \mathbf{0}$, so hat diese Kante keinen Einfluss auf $\mathbb{P}(\boldsymbol{X} = \boldsymbol{x})$!

$$\mathbb{P}(\boldsymbol{X} = \boldsymbol{x}) = \frac{1}{Z(\boldsymbol{\beta})} \exp(\langle \boldsymbol{\beta}, \phi(\boldsymbol{x}) \rangle) = \frac{\prod_{C \in \mathcal{C}(G)} \exp(\langle \boldsymbol{\beta}_C, \phi_C(\boldsymbol{x}_C) \rangle)}{\sum_{\boldsymbol{x}' \in \mathcal{X}} \prod_{C \in \mathcal{C}(G)} \exp(\langle \boldsymbol{\beta}_C, \phi_C(\boldsymbol{x}'_C) \rangle)}$$

Falls alle Cliquenfaktoren in paarweise Funktionen zerfallen:

$$\mathbb{P}(\boldsymbol{X} = \boldsymbol{x}) = \frac{\prod_{\{v,u\} \in E} \exp(\langle \boldsymbol{\beta}_{\{v,u\}}, \phi_{\{v,u\}}(\boldsymbol{x}_{\{v,u\}}) \rangle)}{\sum_{\boldsymbol{x}' \in \mathcal{X}} \prod_{\{v,u\} \in E} \exp(\langle \boldsymbol{\beta}_{\{v,u\}}, \phi_{\{v,u\}}(\boldsymbol{x}'_{\{v,u\}}) \rangle)}$$

Falls $\boldsymbol{\beta}_{\{v,u\}} = \mathbf{0}$ für eine Kante $\{v, u\}$, so ist $\exp(\langle \boldsymbol{\beta}_{\{v,u\}}, \phi_{\{v,u\}}(\boldsymbol{x}_{\{v,u\}}) \rangle) = 1$.

Option2: Graphen mittels Regularisierung

Baobachtung: Falls Parametervektor $\boldsymbol{\beta}_{\{v,u\}}$ einer Kante $= \mathbf{0}$, so hat diese Kante keinen Einfluss auf $\mathbb{P}(\boldsymbol{X} = \boldsymbol{x})$!

$$\mathbb{P}(\boldsymbol{X} = \boldsymbol{x}) = \frac{1}{Z(\boldsymbol{\beta})} \exp(\langle \boldsymbol{\beta}, \phi(\boldsymbol{x}) \rangle) = \frac{\prod_{C \in \mathcal{C}(G)} \exp(\langle \boldsymbol{\beta}_C, \phi_C(\boldsymbol{x}_C) \rangle)}{\sum_{\boldsymbol{x}' \in \mathcal{X}} \prod_{C \in \mathcal{C}(G)} \exp(\langle \boldsymbol{\beta}_C, \phi_C(\boldsymbol{x}'_C) \rangle)}$$

Falls alle Cliquenfaktoren in paarweise Funktionen zerfallen:

$$\mathbb{P}(\boldsymbol{X} = \boldsymbol{x}) = \frac{\prod_{\{v,u\} \in E} \exp(\langle \boldsymbol{\beta}_{\{v,u\}}, \phi_{\{v,u\}}(\boldsymbol{x}_{\{v,u\}}) \rangle)}{\sum_{\boldsymbol{x}' \in \mathcal{X}} \prod_{\{v,u\} \in E} \exp(\langle \boldsymbol{\beta}_{\{v,u\}}, \phi_{\{v,u\}}(\boldsymbol{x}'_{\{v,u\}}) \rangle)}$$

Falls $\boldsymbol{\beta}_{\{v,u\}} = \mathbf{0}$ für eine Kante $\{v, u\}$, so ist $\exp(\langle \boldsymbol{\beta}_{\{v,u\}}, \phi_{\{v,u\}}(\boldsymbol{x}_{\{v,u\}}) \rangle) = 1$.

Option2: Graphen mittels Regularisierung

Baobachtung: Falls Parametervektor $\boldsymbol{\beta}_{\{v,u\}}$ einer Kante $= \mathbf{0}$, so hat diese Kante keinen Einfluss auf $\mathbb{P}(\boldsymbol{X} = \boldsymbol{x})$!

Idee: Minimiere $\ell(\boldsymbol{\beta}; \mathcal{D}) + \lambda \|\boldsymbol{\beta}\|_1$

Links- und rechtsseitige Grenzwerte stimmen am Nullpunkt $(\mathbf{0})$ nicht überein!

$\|\cdot\|_1$ nicht differenzierbar(!) $\rightarrow$ Gradient kann nicht berechnet werden!!

Aber: Proximaler Gradientenabstieg möglich.

Proximaler Gradientenabstieg

Falls Funktion f nicht differenzierbar: Für die Minimierung von

$$F(\boldsymbol{\beta}; \mathcal{D}) = \ell(\boldsymbol{\beta}^t; \mathcal{D}) + f(\boldsymbol{\beta})$$

Nutzen wir anstatt

$$\boldsymbol{\beta}^{t+1} = \boldsymbol{\beta}^t - \eta_t \nabla F(\boldsymbol{\beta}^t; \mathcal{D})$$

jetzt

$$\boldsymbol{\beta}^{t+1} = \text{prox}_f(\boldsymbol{\beta}^t - \eta_t \nabla \ell(\boldsymbol{\beta}^t; \mathcal{D}))$$

Funktion F wird aufgeteilt in den differenzierbaren Teil ℓ und f
Intuition an der Tafel...

Proximaler Gradientenabstieg

Falls Funktion f nicht differenzierbar: Für die Minimierung von

$$F(\boldsymbol{\beta}; \mathcal{D}) = \ell(\boldsymbol{\beta}^t; \mathcal{D}) + f(\boldsymbol{\beta})$$

Nutzen wir anstatt

$$\boldsymbol{\beta}^{t+1} = \boldsymbol{\beta}^t - \eta_t \nabla F(\boldsymbol{\beta}^t; \mathcal{D})$$

jetzt

$$\boldsymbol{\beta}^{t+1} = \text{prox}_f(\boldsymbol{\beta}^t - \eta_t \nabla \ell(\boldsymbol{\beta}^t; \mathcal{D}))$$

Funktion F wird aufgeteilt in den differenzierbaren Teil ℓ und f
Intuition an der Tafel...

Proximaler Gradientenabstieg für l_1-Regularisierung

Formal:

$$\text{prox}_{\lambda\|\cdot\|_1}(\gamma_i) = \min_a \left\{ \|a\|_1 + \frac{1}{2\lambda}\|a - \gamma_i\|_2^2 \right\}$$

Lösung:

$$\text{prox}_{\lambda\|\cdot\|_1}(\gamma_i) = \begin{cases} \gamma_i - \lambda & , \gamma_i > \lambda \\ \gamma_i + \lambda & , \gamma_i < -\lambda \\ 0 & , \text{sonst} \end{cases}$$

Algorithmus zur Bestimmung von G:

(1) Wähle vollständigen Graphen G' (mit allen möglichen Kanten)

(2) Löse $\min_\beta \ell(\beta; \mathcal{D}) + \lambda\|\beta\|_1$ mittels proximalem Gradientenabstieg

(3) Entferne alle Kanten $\{v, u\}$ mit $\beta_{\{v,u\}} = \mathbf{0}$ aus G'

Proximaler Gradientenabstieg für l_1-Regularisierung

Formal:

$$\text{prox}_{\lambda\|\cdot\|_1}(\gamma_i) = \min_a \left\{ \|a\|_1 + \frac{1}{2\lambda}\|a - \gamma_i\|_2^2 \right\}$$

Lösung:

$$\text{prox}_{\lambda\|\cdot\|_1}(\gamma_i) = \begin{cases} \gamma_i - \lambda & , \gamma_i > \lambda \\ \gamma_i + \lambda & , \gamma_i < -\lambda \\ 0 & , \text{sonst} \end{cases}$$

Algorithmus zur Bestimmung von G:

(1) Wähle vollständigen Graphen G' (mit allen möglichen Kanten)

(2) Löse $\min_\beta \ell(\beta; \mathcal{D}) + \lambda\|\beta\|_1$ mittels proximalem Gradientenabstieg

(3) Entferne alle Kanten $\{v, u\}$ mit $\beta_{\{v,u\}} = 0$ aus G'

Proximaler Gradientenabstieg für l_1-Regularisierung

Formal:

$$\text{prox}_{\lambda\|\cdot\|_1}(\gamma_i) = \min_a \left\{ \|a\|_1 + \frac{1}{2\lambda}\|a - \gamma_i\|_2^2 \right\}$$

Lösung:

$$\text{prox}_{\lambda\|\cdot\|_1}(\gamma_i) = \begin{cases} \gamma_i - \lambda & , \gamma_i > \lambda \\ \gamma_i + \lambda & , \gamma_i < -\lambda \\ 0 & , \text{sonst} \end{cases}$$

Algorithmus zur Bestimmung von G:

(1) Wähle vollständigen Graphen G' (mit allen möglichen Kanten)

(2) Löse $\min_{\boldsymbol{\beta}} \ell(\boldsymbol{\beta}; \mathcal{D}) + \lambda\|\boldsymbol{\beta}\|_1$ mittels proximalem Gradientenabstieg

(3) Entferne alle Kanten $\{v, u\}$ mit $\boldsymbol{\beta}_{\{v,u\}} = \mathbf{0}$ aus G'