Chow-Liu

• Goal: find a tree that maximizes the data likelihood

Algorithm

• Compute weight $I(X_i, X_j)$ of each (possible) edge $(X_i, X_j)$

• Find a maximum weight spanning tree (MST)

• Give directions to edges in MST
Chow-Liu: how-to

• Goal: find a tree that maximizes the data likelihood

Algorithm

• Compute weight $I(X_i, X_j)$ of each (possible) edge $(X_i, X_j)$

$$I(X_i, X_j) = \sum_{x_i, x_j} \hat{P}(x_i, x_j) \log \frac{\hat{P}(x_i, x_j)}{\hat{P}(x_i) \hat{P}(x_j)}$$

$$\hat{P}(x_i, x_j) = \frac{\text{Count}(x_i, x_j)}{m}$$

“empirical distribution”

# examples

• e.g. (1) & (3)

$$I(X_1, X_3) = \sum_{x_1=0}^{2} \sum_{x_2=0}^{2} \hat{P}(X_1 = x_1, X_2 = x_2) \log \frac{\hat{P}(X_1 = x_1, X_2 = x_2)}{\hat{P}(X_1 = x_1) \hat{P}(X_2 = x_2)}$$

e.g. $\hat{P}(X_1 = 0, X_2 = 1) = \frac{\text{Count}(X_1 = 0, X_2 = 1)}{m}$

Chow-Liu: how-to

• Goal: find a tree that maximizes the data likelihood

Algorithm

• Compute weight $I(X_i, X_j)$ of each (possible) edge $(X_i, X_j)$

• Find a maximum weight spanning tree (MST)

• tree with the greatest total weight $\sum_{(X_i, X_j) \in E} I(X_i, X_j)$

• greedily add edges, just make sure it’s a tree at every step

• e.g. Kruskal, Prim
Chow-Liu: how-to

• Goal: find a tree that maximizes the data likelihood

Algorithm

• Compute weight $I(X_i, X_j)$ of each (possible) edge $(X_i, X_j)$

• Find a maximum weight spanning tree (MST)
  
  • tree with the greatest total weight $\sum_{(X_i, X_j) \in E} I(X_i, X_j)$

  • greedily add edges, just make sure it’s a tree at every step
Chow-Liu: how-to

- Goal: find a tree that maximizes the data likelihood

**Algorithm**

- Compute weight $I(X_i, X_j)$ of each (possible) edge $(X_i, X_j)$
- Find a maximum weight spanning tree (MST)
  - tree with the greatest total weight $\sum_{(X_i, X_j) \in E} I(X_i, X_j)$
  - greedily add edges, just make sure it’s a tree at every step

```
1
2
3
4

1
2
3
4
```

Chow-Liu: how-to

- Goal: find a tree that maximizes the data likelihood

**Algorithm**

- Compute weight $I(X_i, X_j)$ of each (possible) edge $(X_i, X_j)$
- Find a maximum weight spanning tree (MST)
- Give directions to edges in MST
  - pick your favorite node (e.g. sinus??)
  - draw arrows going away from it (e.g. BFS, DFS)

```
1
2
3
4
```
Chow-Liu: why it works

• Goal: find a tree that maximizes the data likelihood

**Algorithm**

- Compute weight $I(X_i, X_j)$ of each (possible) edge $(X_i, X_j)$
- Find a maximum weight spanning tree (MST)
- Give directions to edges in MST

Just two questions:
1. why can we represent data likelihood as sum of $I(X_i, X_j)$ over edges?
2. why can we pick any direction for edges in the tree?* 

*as long as it’s a tree

1. why can we represent data likelihood as sum of $I(X_i, X_j)$ over edges?
2. why can we pick any direction for edges in the tree?

- data likelihood given (directed) edges
  \[
  \log P(D | G, \theta_G) = \sum_{j=1}^{m} \sum_{i=1}^{n} \log P(x_i | pa X_i)
  \]
- information theoretic quantity
  \[
  \log P(D | G, \theta_G) = m \sum_{i=1}^{n} (I(X_i, Pa X_i) - H(X_i))
  \]
- max only part that matters
  \[
  \arg\max_{G} \log P(D | G, \theta_G) = \arg\max_{G} \sum_{i=1}^{n} I(X_i, Pa X_i)
  \]
- tree! (Pa_Xi = just one other node) => $I(X_i, Pa_Xi) = I(Xi, Xj)$
  \[
  \arg\max_{G} \log P(D | G, \theta_G) = \arg\max_{G} \sum_{(X_i, X_j) \in E} I(X_i, X_j)
  \]
- directed edges? nah. $I(Xi, Xj) = I(Xj, Xi)$
  \[
  I(X_i, X_j) = \sum_{x_i,x_j} \hat{P}(x_i, x_j) \log \frac{\hat{P}(x_i, x_j)}{P(x_i)P(x_j)}
  \]
1. Why can we represent data likelihood as sum of \( I(X_i, X_j) \) over edges?

2. Why can we pick any direction for edges in the tree?

- Data likelihood given (directed) edges
  \[
  \log P(D | G, \theta_G) = \sum_{j=1}^{m} \sum_{i=1}^{n} \log P(x_i | pa_X_i)
  \]

- Information theoretic quantity
  \[
  \log P(D | G, \theta_G) = m \sum_{i=1}^{n} (I(X_i, pa_X_i) - H(X_i))
  \]

- Max only part that matters
  \[
  \arg \max_G \log P(D | G, \theta_G) = \arg \max_G \sum_{i=1}^{n} I(X_i, pa_X_i)
  \]

- Tree! (\( Pa_X_i \) just one other node) \( \Rightarrow \) \( I(X_i, Pa_X_i) = I(X_i, X_j) \)

- Directed edges? Nah. \( I(X_i, X_j) = I(X_j, X_i) \)

- So directions don't matter

- As long as no v-structures

---

**TAN::Tree-Augmented Naive Bayes**

- NB + Chow-Liu

  - Same old Chow-Liu on features, but with \( I(X_i, X_j|c) \) instead of \( I(X_i, X_j) \)

  - Then learn \( P(X_i | Pa(X_i), c) \) as before

  - **Remember** this algorithm for the future
the usual difficulties

- In general, NP-hard to learn structure with #parents > 1
  try e.g.
  - BIC score: approximation of Bayesian score
    maximizing still NP-hard
    "regularization"

- Trees - “easy” to learn:
  - one parent - no v-structures
    can do this greedy search with completely uncoupled scores

Announcing

- no recitation next week - happy thanksgiving!
  - better sleep...!