Lecture 24. A Case Study of Ensembling Unreliable Sources

COMP90051 Statistical Machine Learning

Semester 1, 2021
Lecturer: Yuan Li

THE UNIVERSITY OF MELBOURNE

A truth inference problem...

x_{ij}	j=1	j=2	j=3	j=4	j=5		y_i
i=1	1	1	1	1	1		?
i=2	1	1	0	0	1		?
i=3	0	0	0	1	0		?
i=4	0	1	0	0	1		?
i=5	1	1	1	0	1	→	?
i=6	0	1	1	1	1		?
i=7	1	0	0	0	1		?
i=8	0	1	0	0	1		?
i=9	1	1	0	1	1		?
…	…	…	…	…	…		…

- to build a dataset for binary classification
- 1000 items
- 5 workers
- everyone has labeled everything
- how to infer the ground truth?

Notations

x_{ij}	j=1	j=2	j=3	j=4	j=5		y_i
i=1	1	1	1	1	1		?
i=2	1	1	0	0	1		?
i=3	0	0	0	1	0		?
i=4	0	1	0	0	1		?
i=5	1	1	1	0	1	→	?
i=6	0	1	1	1	1		?
i=7	1	0	0	0	1		?
i=8	0	1	0	0	1		?
i=9	1	1	0	1	1		?
…	…	…	…	…	…		…

- 1000 items, i=1…1000
- 5 workers, j=1,2,3,4,5
- x_{ij} - worker j's label to item i
- y_i - true label of item i

Consider 3 scenarios

- 1000 items, 5 workers
- every worker has labeled every item -> 5000 labels

- High resource – have many gold labels, e.g., 500
- No resource – don't have any gold labels
- Low resource – have a few, e.g., 5

Consider 3 scenarios

- 1000 items, 5 workers
- every worker has labeled every item -> 5000 labels

- High resource – have many gold labels, e.g., 500
- No resource – don't have any gold labels
- Low resource – have a few, e.g., 5

High resource

x_{ij}	j=1	j=2	j=3	j=4	j=5
1	1	1	1	1	1
2	1	1	0	0	1
3	0	0	0	1	0
4	0	1	0	0	1
5	1	1	1	0	1
6	0	1	1	1	1
7	1	0	0	0	1
8	0	1	0	0	1
9	1	1	0	1	1
…	…	…	…	…	…
500	0	0	0	0	0

&

y_i
1
1
0
0
1
1
0
0
1
…
0

- 500 items with gold labels as training set
- treat each worker as a feature
- every row as a 5-d feature vector
- train a binary classifier

High resource

x_{ij}	j=1	j=2	j=3	j=4	j=5		y_i
501	0	0	1	1	0		?
502	1	1	0	1	0		?
503	1	1	1	1	0		?
504	0	0	1	1	0		?
505	1	1	0	0	1	→	?
506	0	0	1	0	0		?
507	1	1	1	1	1		?
508	1	0	1	1	1		?
509	0	0	0	0	0		?
…	…	…	…	…	…		…
1000	1	1	0	1	1		?

- the rest as test set
- make predictions

Choice of binary classifiers?

x_{ij}	j=1	j=2	j=3	j=4	j=5
1	1	1	1	1	1
2	1	1	0	0	1
3	0	0	0	1	0
4	0	1	0	0	1
5	1	1	1	0	1
6	0	1	1	1	1
7	1	0	0	0	1
8	0	1	0	0	1
9	1	1	0	1	1
…	…	…	…	…	…
500	0	0	0	0	0

&

y_i
1
1
0
0
1
1
0
0
1
…
0

- Logistic regression
- SVM
- Neural networks
- Random forests
- Boosting trees
- Also, Naïve Bayes!

Choice of binary classifiers?

x_{ij}	j=1	j=2	j=3	j=4	j=5		y_i
1	1	1	1	1	1		1
2	1	1	0	0	1		1
3	0	0	0	1	0		0
4	0	1	0	0	1		0
5	1	1	1	0	1	&	1
6	0	1	1	1	1		1
7	1	0	0	0	1		0
8	0	1	0	0	1		0
9	1	1	0	1	1		1
…	…	…	…	…	…		…
500	0	0	0	0	0		0

- Logistic regression
- SVM
- Neural networks
- Random forests
- Boosting trees
- Also, Naïve Bayes!
  - *generative model, others are discriminative*

Compare NB with LR

- Logistic regression
  - models $P(y|x, w, b)$, learns weights (w) and bias (b)
  - $\sum_{i=1}^{5} w_i x_i + b > 0 \rightarrow y = 1$, otherwise, $y = 0$
- Naïve Bayes
  - $P(x, y|\theta) = P(x_1, x_2, x_3, x_4, x_5|y, \theta)P(y)$
  $= P(x_1|y, \theta)P(x_2|y, \theta)P(x_3|y, \theta)P(x_4|y, \theta)P(x_5|y, \theta)P(y)$
    - *conditional independence assumption*
  - learns confusion matrices (θ) and class distribution $P(y)$
  - $P(x, y = 1|\theta) > P(x, y = 0|\theta) \rightarrow y = 1$
    - otherwise, $y = 0$

What NB learns...

- Class distribution
  - $P(y = 0) = \#\{y = 0\}/500$
  - $P(y = 1) = \#\{y = 1\}/500$
- Confusion matrix for each worker

$$\begin{bmatrix} P(x = 0|y = 0) & P(x = 1|y = 0) \\ P(x = 0|y = 1) & P(x = 1|y = 1) \end{bmatrix}$$

  - $P(x = 0|y = 0) = \#\{x = 0, y = 0\}/\#\{y = 0\}$
  - $P(x = 1|y = 0) = 1 - P(x = 0|y = 0)$
- Sometimes called 2-coin model

Use NB to make prediction for 11000

Confusion matrix	
Worker 1	$\begin{bmatrix} 0.9 & 0.1 \\ 0.2 & 0.8 \end{bmatrix}$
Worker 2	$\begin{bmatrix} 0.8 & 0.2 \\ 0.3 & 0.7 \end{bmatrix}$
Worker 3	$\begin{bmatrix} 0.7 & 0.3 \\ 0.4 & 0.6 \end{bmatrix}$
Worker 4	$\begin{bmatrix} 0.6 & 0.4 \\ 0.3 & 0.7 \end{bmatrix}$
Worker 5	$\begin{bmatrix} 0.7 & 0.3 \\ 0.2 & 0.8 \end{bmatrix}$
Class distribution	
$P(y=0)=0.5$	$P(y=1)=0.5$

$$\begin{bmatrix} P(x=0|y=0) & P(x=1|y=0) \\ P(x=0|y=1) & P(x=1|y=1) \end{bmatrix}$$

- P(11000|y=0)=0.1*0.2*0.7*0.6*0.7=0.00588
- P(11000|y=1)=0.8*0.7*0.4*0.3*0.2=0.01344
- So…
- P(11000|y=1) is larger
- More likely the true label is 1

Consider 3 scenarios

- 1000 items, 5 workers
- every worker has labeled every item -> 5000 labels

- High resource – have many gold labels, e.g., 500
- No resource – don’t have any gold labels
- Low resource – have a few, e.g., 5

No resource

x_{ij}	j=1	j=2	j=3	j=4	j=5		y_i
1	1	1	1	1	1		?
2	1	1	0	0	1		?
3	0	0	0	1	0		?
4	0	1	0	0	1		?
5	1	1	1	0	1	→	?
6	0	1	1	1	1		?
7	1	0	0	0	1		?
8	0	1	0	0	1		?
9	1	1	0	1	1		?
…	…	…	…	…	…		…
1000	1	1	0	1	1		?

- no gold label
- can't use supervised models
- majority voting?

Majority Voting

- sum = x1+x2+x3+x4+x5
- if sum=3,4,5 -> predicted label = 1
- if sum=0,1,2 -> predicted label = 0

- But workers are not equally accurate…

Use NB to make prediction for 11000

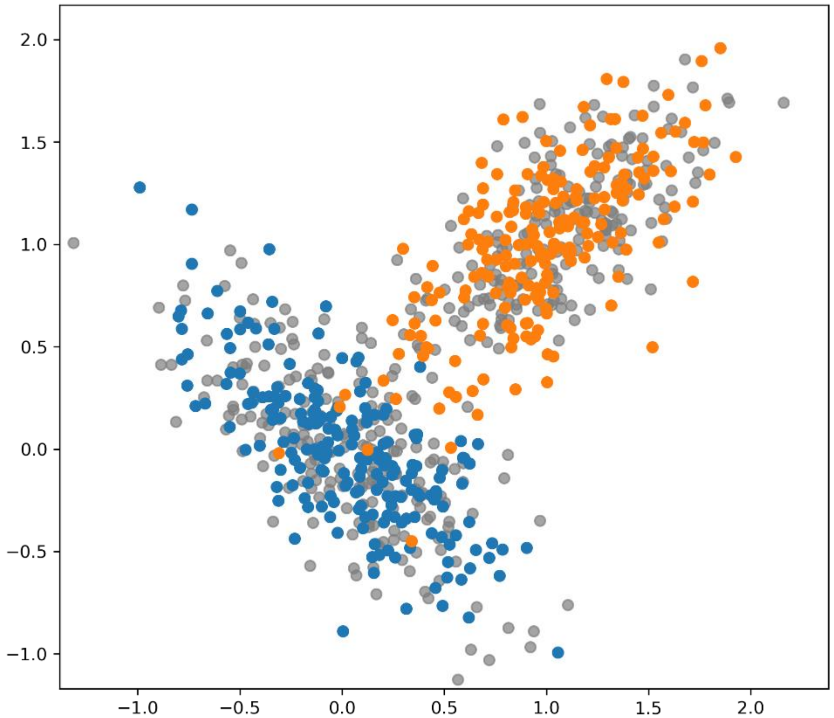
Confusion matrix	
Worker 1	$\begin{bmatrix} 0.9 & 0.1 \\ 0.2 & 0.8 \end{bmatrix}$
Worker 2	$\begin{bmatrix} 0.8 & 0.2 \\ 0.3 & 0.7 \end{bmatrix}$
Worker 3	$\begin{bmatrix} 0.7 & 0.3 \\ 0.4 & 0.6 \end{bmatrix}$
Worker 4	$\begin{bmatrix} 0.6 & 0.4 \\ 0.3 & 0.7 \end{bmatrix}$
Worker 5	$\begin{bmatrix} 0.7 & 0.3 \\ 0.2 & 0.8 \end{bmatrix}$
Class distribution	
$P(y=0)=0.5$	$P(y=1)=0.5$

$$\begin{bmatrix} P(x=0|y=0) & P(x=1|y=0) \\ P(x=0|y=1) & P(x=1|y=1) \end{bmatrix}$$

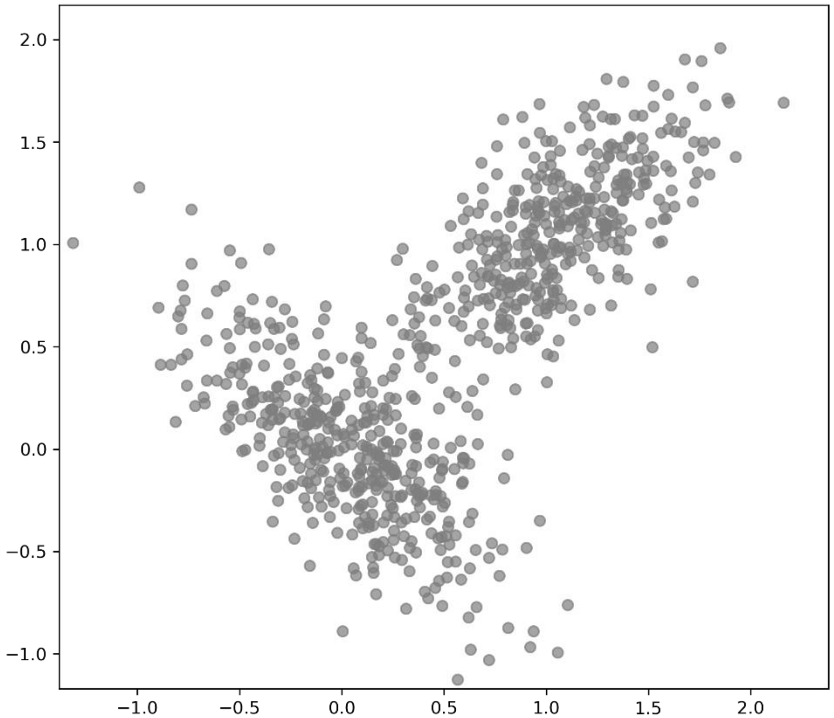
- P(11000|y=0)=0.1*0.2*0.7*0.6*0.7=0.00588
- P(11000|y=1)=0.8*0.7*0.4*0.3*0.2=0.01344
- So…
- P(11000|y=1) is larger
- More likely the true label is 1

1000 points, binary classification

High resource, 500 gold labels

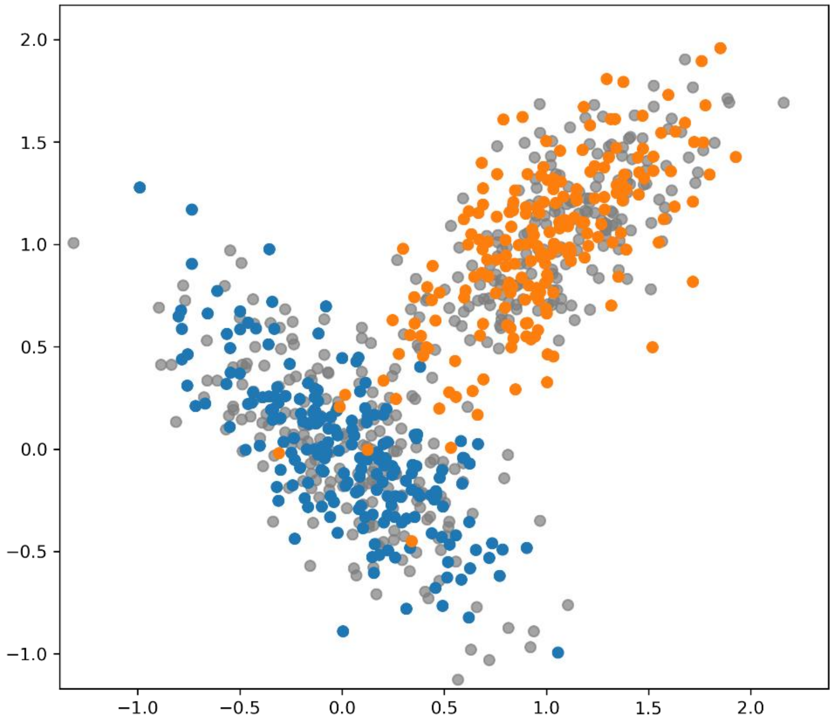


No resource, 0 gold label

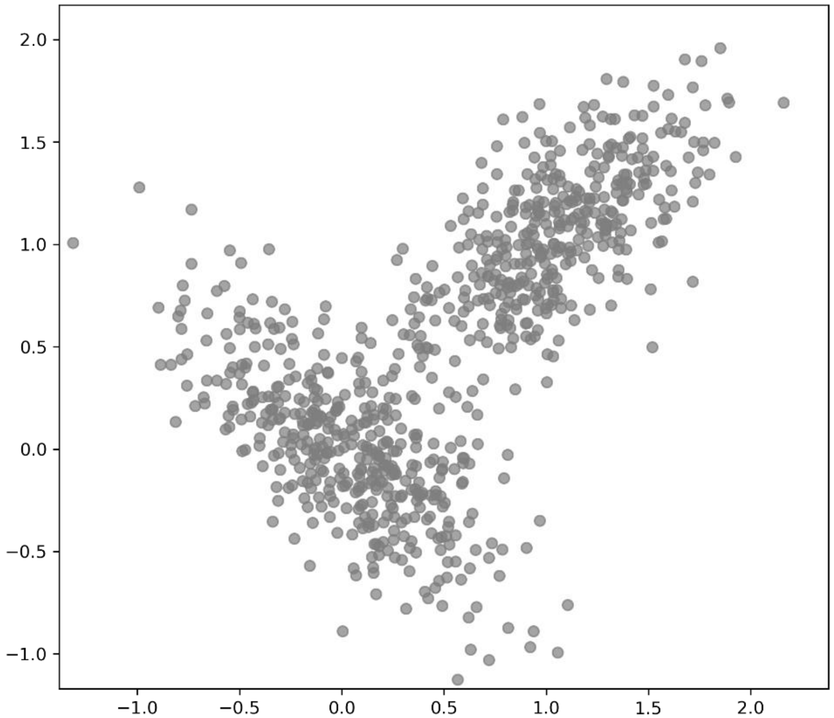


GMM can work in both cases

High resource, 500 gold labels



No resource, 0 gold label



Benefits of generative models

- Optimizing discriminative models requires gold labels
  - $\theta^* = \text{argmax}_\theta \, P(Y|X,\theta)$ requires Y
- Generative models define the joint distribution $P(X,Y|\theta) = P(X|Y,\theta)P(Y|\theta)$
  - can work when Y is known
    - $\theta^* = \text{argmax}_\theta \, P(X,Y|\theta)$
  - can also work when Y is unknown, by marginalization
    - $P(X|\theta) = \sum_Y P(X,Y|\theta)$
    - $\theta^* = \text{argmax}_\theta \, P(X|\theta)$
- So, unsupervised Naïve Bayes?

NB vs. GMM

- Naïve Bayes
  - $P(x, y|\theta) = P(x_1, x_2, x_3, x_4, x_5|y, \theta)P(y)$
  $= P(x_1|y, \theta)P(x_2|y, \theta)P(x_3|y, \theta)P(x_4|y, \theta)P(x_5|y, \theta)P(y)$
- GMM
  - $P(x, y|\mu, \Sigma) = P(x_1, x_2|y, \mu, \Sigma)P(y) = \mathcal{N}\left(x_1, x_2\middle|\mu_y, \Sigma_y\right)P(y)$
- GMM learns clusters
- What does NB learn?

Naïve Bayes as clustering

- Imagine 2 clusters with 00000 and 11111 as centers
- every $(x_1, x_2, x_3, x_4, x_5)$ is generated from one of them
  * $P(x_1, x_2, x_3, x_4, x_5 | y = c)$ defines the probability that $(x_1, x_2, x_3, x_4, x_5)$ is generated from cluster c
- Think about the “similarity” between 00000 and
  * 00000, 10000, 11000, 11100, 11110, 11111
- As it moves away from the center 00000, the probability, or similarity, goes down

Similarity

Confusion matrix	
Worker 1	$\begin{bmatrix} 0.9 & 0.1 \\ 0.2 & 0.8 \end{bmatrix}$
Worker 2	$\begin{bmatrix} 0.8 & 0.2 \\ 0.3 & 0.7 \end{bmatrix}$
Worker 3	$\begin{bmatrix} 0.7 & 0.3 \\ 0.4 & 0.6 \end{bmatrix}$
Worker 4	$\begin{bmatrix} 0.6 & 0.4 \\ 0.3 & 0.7 \end{bmatrix}$
Worker 5	$\begin{bmatrix} 0.7 & 0.3 \\ 0.2 & 0.8 \end{bmatrix}$
Class distribution	
P(y=0)=0.5	P(y=1)=0.5

$$\begin{bmatrix} P(x=0|y=0) & P(x=1|y=0) \\ P(x=0|y=1) & P(x=1|y=1) \end{bmatrix}$$

- P(00000|y=0)=0.9*0.8*0.7*0.6*0.7=0.21168
- P(10000|y=0)=0.1*0.8*0.7*0.6*0.7=0.02352
- P(11000|y=0)=0.00588
- P(11100|y=0)=0.00252
- P(11110|y=0)=0.00168
- P(11111|y=0)=0.00072

NB vs. GMM

- Naïve Bayes
  - $P(x, y|\theta) = P(x_1, x_2, x_3, x_4, x_5|y, \theta)P(y)$
  $= P(x_1|y, \theta)P(x_2|y, \theta)P(x_3|y, \theta)P(x_4|y, \theta)P(x_5|y, \theta)P(y)$
- GMM
  - $P(x, y|\mu, \Sigma) = P(x_1, x_2|y, \mu, \Sigma)P(y) = \mathcal{N}\left(x_1, x_2\middle|\mu_y, \Sigma_y\right)P(y)$
- GMM learns clusters
- What does NB learn?
  - NB also learns clusters

Formally, unsupervised NB

- θ – unknown parameters
- $P(X, Y|\theta)$ – joint distribution
- We only have observed X but not Y
- To learn θ, we first marginalize Y to get
  - $P(X|\theta) = \sum_Y P(X, Y|\theta)$
- Then $\theta^* = \text{argmax}_\theta P(X|\theta)$
- Use θ^* to make predictions $P(Y|X, \theta^*)$

Consider 3 scenarios

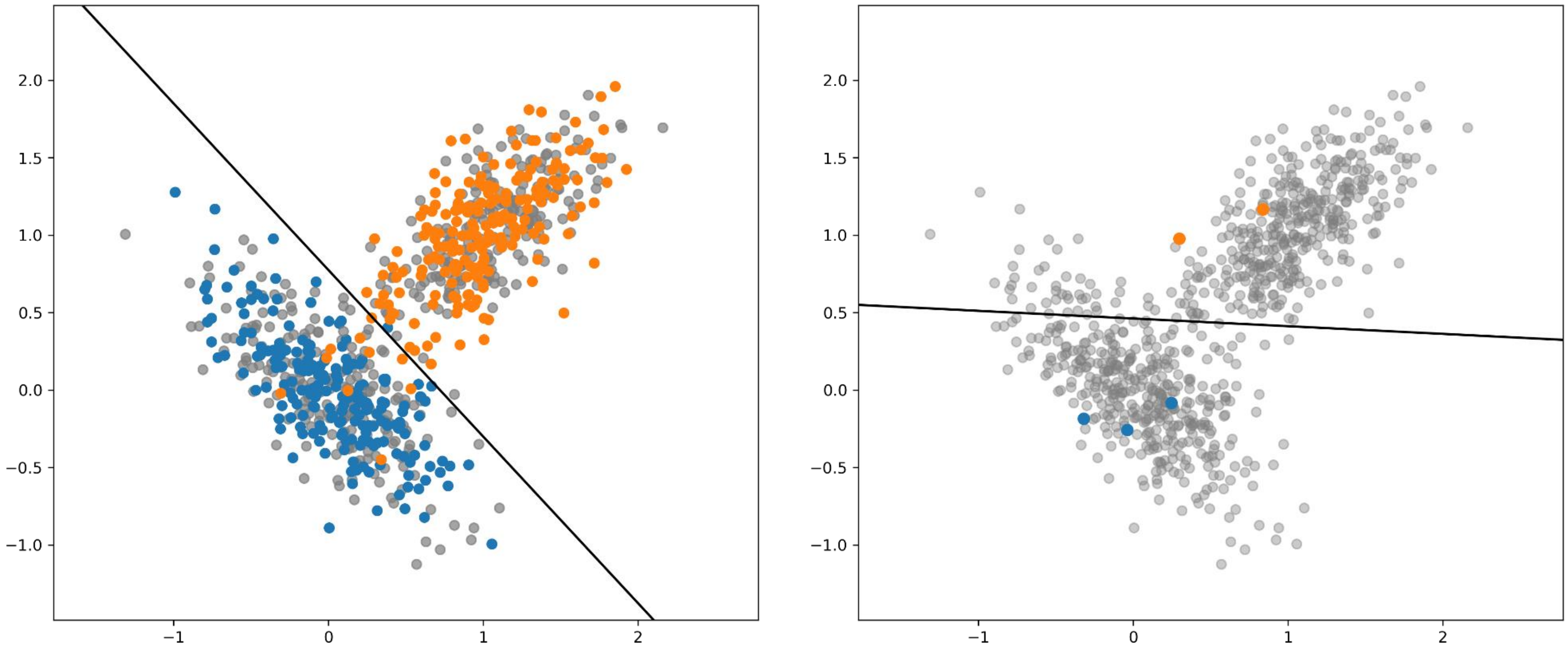
- 1000 items, 5 workers
- every worker has labeled every item -> 5000 labels

- High resource – have many gold labels, e.g., 500
- No resource – don't have any gold labels
- Low resource – have a few, e.g., 5

Can we do the same?

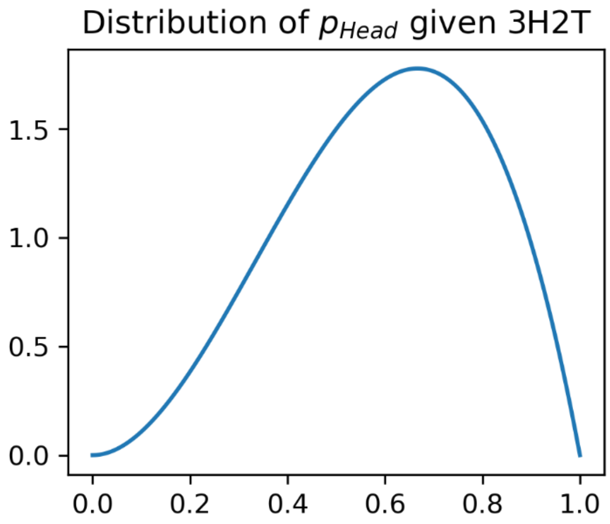
- High resource
  - learn θ^* from gold data
  - $P(Y|X,\theta^*)$ → predictions
- Low resource
  - learn θ^* from gold data
  - $P(Y|X,\theta^*)$ → predictions

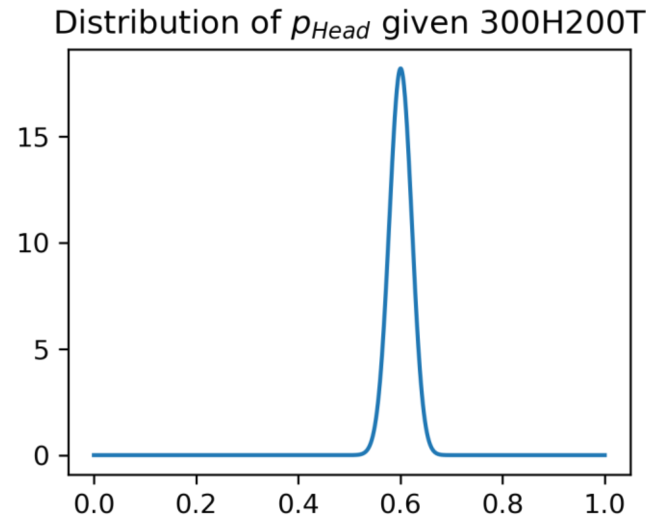
5 is insufficient for reliable estimate



Distribution vs. Point estimate

- distribution carries uncertainty (lost in point estimate)
- 3 heads + 2 tails vs. 300 heads + 200 tails





~~Can we do the same?~~

- High resource
  - learn θ^* from gold data
  - $P(Y|X, \theta^*)$ → predictions
- Low resource
  - ~~learn θ^* from gold data~~
  - ~~$P(Y|X, \theta^*)$ → predictions~~
  - learn a prior from gold $P(\theta) = P\left(\theta \middle| X_{gold}, Y_{gold}\right)$
  - $\theta^* = \mathrm{argmax}_\theta\, P(X|\theta)P(\theta)$
  - $P(Y|X, \theta^*)$ → predictions

Summary

- High resource (any supervised model)
  - learn θ^* from gold data
- No resource (unsupervised NB)
  - $\theta^* = \text{argmax}_\theta \, P(X|\theta)$ MLE
- Low resource (unsupervised NB)
  - learn a prior from gold $P(\theta) = P(\theta|X_{gold}, Y_{gold})$
  - $\theta^* = \text{argmax}_\theta \, P(X|\theta)P(\theta)$ MAP
- $P(Y|X, \theta^*)$ → predictions

Can set a prior to encode our belief

- High resource (any supervised model)
  - learn θ^* from gold data
- No resource (unsupervised NB)
  - $\theta^* = \text{argmax}_\theta\, P(X|\theta)P(\theta)$ MAP
- Low resource (unsupervised NB)
  - learn a prior from gold $P(\theta) = P\left(\theta \middle| X_{gold}, Y_{gold}\right)$
  - $\theta^* = \text{argmax}_\theta\, P(X|\theta)P(\theta)$ MAP
- $P(Y|X, \theta^*)$ → predictions

Available algorithms

- Gradient descent
  - general optimization method, widely used
- EM algorithm
  - not suitable for general purpose, like neural networks
  - good at marginalization objective
    - $P(X|\theta) = \sum_Y P(X,Y|\theta)$

$$\theta^* = \mathrm{argmax}_\theta\, P(X|\theta)P(\theta)$$

- Gradient descent
  - $\theta^* = \mathrm{argmin}_\theta - \log P(X|\theta) - \log P(\theta)$
- EM algorithm
- Both should work for either MLE or MAP, but
  - parameters have constraints, e.g., sum up to 1, non-negative
  - gradient descent is good for unconstrainted optimization
  - not trivial to apply it to constrained optimization
- So, EM is better

Learning process

- High resource
- gold labels -> θ^*
- θ^* -> predictions

- No resource/Low resource
- initialize θ^0
- for t=0,1,2,3,...
  - θ^t -> predictions
    - known as pseudo labels
  - pseudo labels -> θ^{t+1}
- until converge
- θ^* -> predictions

Problem: we still have point estimate

- High resource (any supervised model)
  - learn θ^* from gold data
- No resource (unsupervised NB)
  - $\theta^* = \text{argmax}_\theta \, P(X|\theta)P(\theta)$
- Low resource (unsupervised NB)
  - learn a prior from gold $P(\theta) = P(\theta | X_{gold}, Y_{gold})$
  - $\theta^* = \text{argmax}_\theta \, P(X|\theta)P(\theta)$
- $P(Y|X, \theta^*)$ → predictions

Replace θ^* with posterior of θ

- High resource (any supervised model)
  - learn θ^* from gold data, $P(Y|X,\theta^*)$ → predictions
- No resource (unsupervised NB)
  - $P(\theta|X) = P(X|\theta)P(\theta)/\int P(X|\theta)P(\theta)\mathrm{d}\theta$
- Low resource (unsupervised NB)
  - learn a prior from gold $P(\theta) = P\left(\theta \middle| X_{gold}, Y_{gold}\right)$
  - $P(\theta|X) = P(X|\theta)P(\theta)/\int P(X|\theta)P(\theta)\mathrm{d}\theta$
- $P(Y|X) = \int P(Y|X,\theta)P(\theta|X)\,\mathrm{d}\theta$ → predictions

Full-Bayesian solution

- High resource (any supervised model)
  - not necessary, point estimate & posterior are very close
- No resource/Low resource
  - Manually set the prior $P(\theta)$ or learn it from gold data
  - get posterior $P(\theta|X) = \frac{P(X|\theta)P(\theta)}{\int P(X|\theta)P(\theta)\mathrm{d}\theta}$
  - $P(Y|X) = \int P(Y|X,\theta)P(\theta|X)\,\mathrm{d}\theta$ → predictions
- Challenge
  - Denominator is often intractable to calculate
  - Exception: Bayesian linear regression has analytical solution

Two approximate inference methods

- Exact inference
  - $P(\theta|X) = \frac{P(X|\theta)P(\theta)}{\int P(X|\theta)P(\theta)\mathrm{d}\theta}$
  - $P(Y|X) = \int P(Y|X,\theta)P(\theta|X)\,\mathrm{d}\theta$ → predictions
- Variational inference
  - Find a simpler distribution $q(\theta) \approx P(\theta|X)$
  - $P(Y|X) \approx \int P(Y|X,\theta)q(\theta)\,\mathrm{d}\theta$ → predictions
- Sampling
  - Draw S samples from $P(\theta|X)$, $\{\theta_1, \theta_2, \dots, \theta_S\}$
  - $P(Y|X) \approx \frac{1}{S}\sum_{i=1}^{S} P(Y|X,\theta_i)$ → predictions

Full-Bayesian vs. Point estimate

- Point estimate
  - $\theta^* = \text{argmax}_\theta P(X|\theta)P(\theta)$
  - $P(Y|X,\theta^*)$ → predictions

- Exact inference
  - $P(\theta|X) = \frac{P(X|\theta)P(\theta)}{\int P(X|\theta)P(\theta)\mathrm{d}\theta}$
  - $P(Y|X) = \int P(Y|X,\theta)P(\theta|X)\,\mathrm{d}\theta$ → predictions

- Variational inference
  - Find a simpler distribution $q(\theta) \approx P(\theta|X)$
  - $P(Y|X) \approx \int P(Y|X,\theta)q(\theta)\,\mathrm{d}\theta$ → predictions

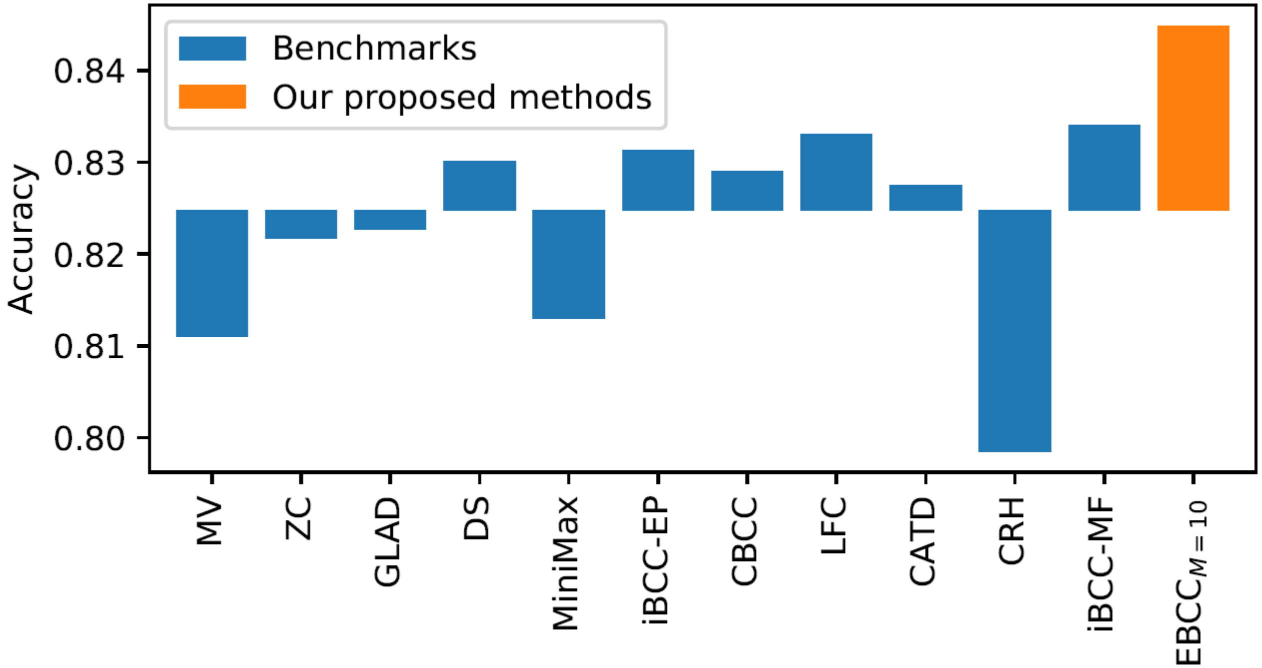
- Sampling
  - Draw S samples from $P(\theta|X)$, $\{\theta_1, \theta_2, \dots, \theta_S\}$
  - $P(Y|X) \approx \frac{1}{S}\sum_{i=1}^{S} P(Y|X,\theta_i)$ → predictions

Extensions

- Full-Bayesian via Sampling (MCMC)
  - Bayesian Classifier Combination (BCC)
    - Kim, Hyun-Chul, and Zoubin Ghahramani. "*Bayesian classifier combination*." In Artificial Intelligence and Statistics, pp. 619-627. PMLR, 2012.
- Our latest work
  - Enhanced BCC (EBCC)
    - Li, Yuan, Benjamin Rubinstein, and Trevor Cohn. "*Exploiting worker correlation for label aggregation in crowdsourcing*." In International Conference on Machine Learning, pp. 3886-3895. PMLR, 2019.

EBCC

- #classes not necessarily equals #clusters
  * Let each class learn M clusters, e.g., M=10
- fit data better
- more flexible



Conclusion

- 1000 items, 5 workers, 5000 labels, 3 scenarios
- discriminative vs. generative, e.g., LR vs. NB
- unsupervised NB, GMM vs. NB, NB as clustering
- low resource vs. high resource
  - point estimate vs. distribution -> full-Bayesian inference
- EM vs. gradient descent for MLE/MAP
- approximate full-Bayesian inference
  - variational inference, sampling

- Thanks!