6: Learning Latent Representations

CS1420: Machine Learning

Michael L. Littman
Spring 2018

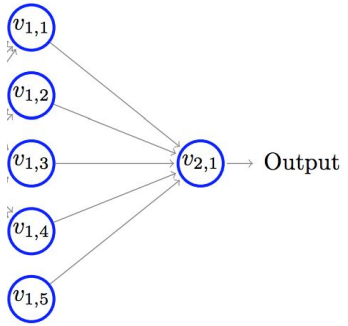
http://www.cs.huji.ac.il/~shais/UnderstandingMachineLearning/ . Sections 24.4.

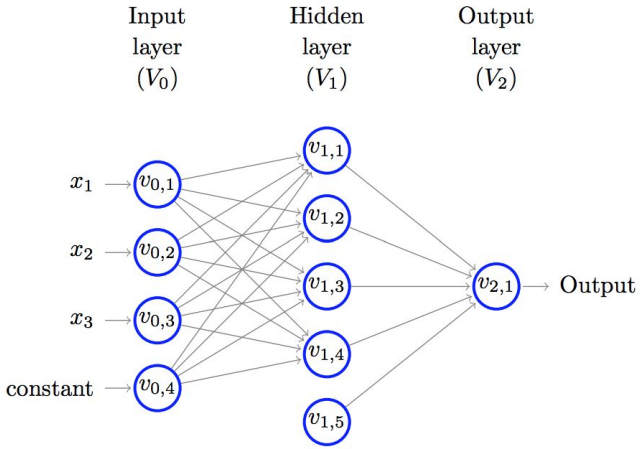
One Vs. Two-Layer NN

Book on gradient descent: “We start with an initial value of $\mathbf{w}$ (say, $\mathbf{w}(1) = 0$).”

Later: “...here we initialize $\mathbf{w}$ to be a randomly chosen vector with values close to zero.”

Why? (A) To show either approach can be used.
(B) To get diverse solutions to avoid local minima.
(C) Deeper networks magnify weight differences.
(D) All hidden neurons will have the same weights.
(E) The zeroes are only used in the analysis.





Internal Computations for Latent Values

Backprop needs to “make up” an internal computation that:

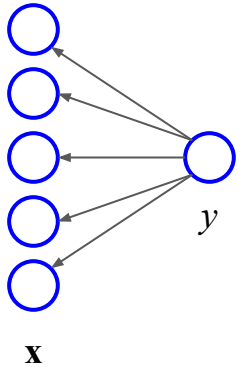
1. can be made based on the inputs, and
2. will help produce valuable outputs.

The values needed are “latent” (hidden, concealed).

Local search from random values is the only game in town for finding them.

(Spectral methods are arguably a second approach.)

Back to Naive Bayes

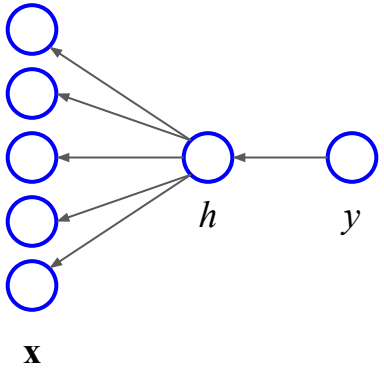


Output randomly chosen. Input features set conditioned on output. Choose most likely output given input.

Easy to train: Pr(input|output), count in data.

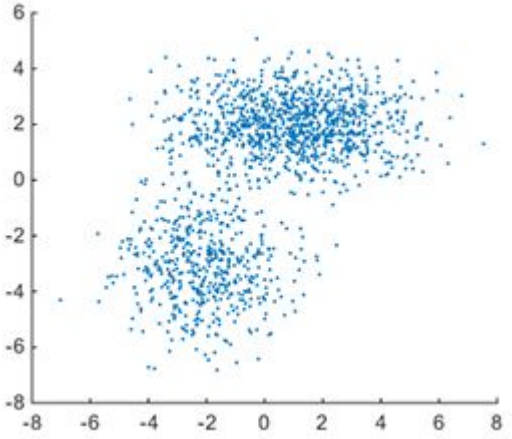
Can make a latent version:

- Output chosen randomly: b^y.
- Latent value set conditioned on output: b^{jy}.
- Input features set conditioned on latent: b^{ij}.



How train?

K Gaussians



We'll look at a related problem first.

$\mathcal{X} = \mathcal{R}^d$. $\mathbf{x} \in \mathcal{X}$ generated by:

1. Choose y with probability c_y for $y \in \{1,...,k\}$.
2. Choose $\mathbf{x}$ from the normal distribution associated with y:
   $\Pr(\mathbf{x}|y) = 1/(2\pi)^{d/2} \exp(-\frac{1}{2} (\mathbf{x}-\mu_y)^T (\mathbf{x}-\mu_y))$.

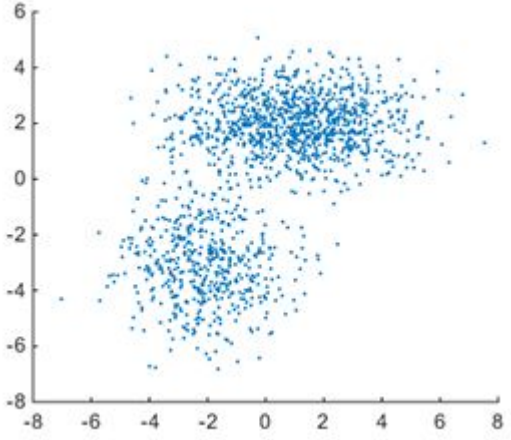
Let θ be the parameters of the model: c_y, μ_y.

Marginalize over y to get density over $\mathcal{X}$.

Am I Your Density?

Which of these formulas is a valid expression representing the probability density? List all digits corresponding to valid formulas, in increasing order.

1. $\sum_{y \in \{1,\ldots,k\}} \Pr(\mathbf{x}, y|\theta)$
2. $\sum_{y \in \{1,\ldots,k\}} \Pr(\mathbf{x}|\theta, y)$
3. $\sum_{y \in \{1,\ldots,k\}} \Pr(y|\theta, \mathbf{x})$
4. $\sum_{y \in \{1,\ldots,k\}} \Pr(\mathbf{x}|\theta, y)\Pr(y|\theta)$
5. $\sum_{y \in \{1,\ldots,k\}} \Pr(\mathbf{x}|\theta)\Pr(y|\theta)$
6. $\sum_{y \in \{1,\ldots,k\}} \Pr(\mathbf{x}|\theta)\Pr(y|\theta, \mathbf{x})$
7. $\Pr(\mathbf{x}|\theta)$
8. $\sum_{y \in \{1,\ldots,k\}} \Pr(\mathbf{x}|\theta, y)/\Pr(y|\theta)$
9. $\sum_{y \in \{1,\ldots,k\}} \Pr(\mathbf{x}, y|\theta)\Pr(\theta)$



How Find Maximum Log Likelihood Parameter?

$\text{argmax}_\theta\, L(\theta) = \text{argmax}_\theta \sum_{\mathbf{x} \in S} \log(\sum_{y \in \{1,\ldots,k\}} \Pr(\mathbf{x},y|\theta))$

Observations:

1. If we knew which y value generated each $\mathbf{x}$, the problem is easier:

$\text{argmax}_\theta\, L(\theta) = \text{argmax}_\theta \sum_{(\mathbf{x},y) \in S} \log(\Pr(\mathbf{x},y|\theta))$: Regular parameter estimation.

2. If we knew θ, knowing which y value generated each $\mathbf{x}$ is easy:

$\Pr(y|\mathbf{x},\theta) = \Pr(\mathbf{x}|y,\theta)\ \Pr(y|\theta) / \Pr(\mathbf{x}|\theta)$: Bayes rule and formulas from previous slide.

Something Else to Optimize

Log likelihood: $L(\theta) = \sum_{i \in \{1,\ldots,m\}} \log(\sum_{y \in \{1,\ldots,k\}} \Pr(\mathbf{x}_i, y|\theta))$

Let Q be an mxk matrix where the rows sum to 1 and define

$$G(Q,\theta) = \sum_{i \in \{1,\ldots,m\}} \sum_{y \in \{1,\ldots,k\}} Q_{iy} \log(\Pr(\mathbf{x}_i, y|\theta)) - \sum_{i \in \{1,\ldots,m\}} \sum_{y \in \{1,\ldots,k\}} Q_{iy} \log(Q_{iy})$$

Nice properties of G:

1. Sum is outside the log now, easier to maximize.
2. If $Q_{iy} = \Pr(y|\mathbf{x}_i, \theta)$, $G(Q,\theta) = L(\theta)$, so finding θ to maximize G maximizes L.
3. For any fixed θ, G is maximized by setting $Q_{iy} = \Pr(y|\mathbf{x}_i, \theta)$.

Proof of Property 2 of G

$G(Q,\theta) = \sum_{i \in \{1,\ldots,m\}} \sum_{y \in \{1,\ldots,k\}} Q_{iy} \log(\Pr(\mathbf{x}_i,y|\theta)) - \sum_{i \in \{1,\ldots,m\}} \sum_{y \in \{1,\ldots,k\}} Q_{iy} \log(Q_{iy})$ G,Q defns

$= \sum_{i \in \{1,\ldots,m\}} \sum_{y \in \{1,\ldots,k\}} \Pr(y|\mathbf{x}_i, \theta) \log(\Pr(\mathbf{x}_i,y|\theta)) - \sum_{i \in \{1,\ldots,m\}} \sum_{y \in \{1,\ldots,k\}} \Pr(y|\mathbf{x}_i, \theta) \log(\Pr(y|\mathbf{x}_i, \theta))$

$= \sum_{i \in \{1,\ldots,m\}} \sum_{y \in \{1,\ldots,k\}} (\Pr(y|\mathbf{x}_i, \theta) \log(\Pr(\mathbf{x}_i,y|\theta)) - \Pr(y|\mathbf{x}_i, \theta) \log(\Pr(y|\mathbf{x}_i, \theta)))$ same sum

$= \sum_{i \in \{1,\ldots,m\}} \sum_{y \in \{1,\ldots,k\}} \Pr(y|\mathbf{x}_i, \theta) (\log(\Pr(\mathbf{x}_i,y|\theta)) - \log(\Pr(y|\mathbf{x}_i, \theta)))$ factor out

$= \sum_{i \in \{1,\ldots,m\}} \sum_{y \in \{1,\ldots,k\}} \Pr(y|\mathbf{x}_i, \theta) \log(\Pr(\mathbf{x}_i,y|\theta) / \Pr(y|\mathbf{x}_i, \theta))$ diff of logs

$= \sum_{i \in \{1,\ldots,m\}} \sum_{y \in \{1,\ldots,k\}} \Pr(y|\mathbf{x}_i, \theta) \log(\Pr(\mathbf{x}_i|\theta))$ cond prob

$= \sum_{i \in \{1,\ldots,m\}} \log(\Pr(\mathbf{x}_i|\theta)) \sum_{y \in \{1,\ldots,k\}} \Pr(y|\mathbf{x}_i, \theta)$ factor

Proof of Property 3 of G

$G(Q,\theta) = \sum_{i \in \{1,\ldots,m\}} \sum_{y \in \{1,\ldots,k\}} Q_{iy} \log(\Pr(\mathbf{x}_i,y|\theta)) - \sum_{i \in \{1,\ldots,m\}} \sum_{y \in \{1,\ldots,k\}} Q_{iy} \log(Q_{iy})$ G defn

$= \sum_{i \in \{1,\ldots,m\}} \sum_{y \in \{1,\ldots,k\}} (Q_{iy} \log(\Pr(\mathbf{x}_i,y|\theta)) - Q_{iy} \log(Q_{iy}))$ same sum

$= \sum_{i \in \{1,\ldots,m\}} \sum_{y \in \{1,\ldots,k\}} Q_{iy} (\log(\Pr(\mathbf{x}_i,y|\theta)) - \log(Q_{iy}))$ factor out

$= \sum_{i \in \{1,\ldots,m\}} \sum_{y \in \{1,\ldots,k\}} Q_{iy} (\log(\Pr(\mathbf{x}_i,y|\theta) / Q_{iy})$ diff of logs

$\leq \sum_{i \in \{1,\ldots,m\}} \log(\sum_{y \in \{1,\ldots,k\}} Q_{iy} \Pr(\mathbf{x}_i,y|\theta) / Q_{iy})$ Jensen!

$= \sum_{i \in \{1,\ldots,m\}} \log(\sum_{y \in \{1,\ldots,k\}} \Pr(\mathbf{x}_i,y|\theta))$ cancel

$= L(\theta).$ defn of L

For all choices of Q, $G(Q,\theta) \leq L(\theta)$. But, Property 2 shows that setting $Q_{iy} = \Pr(y|\mathbf{x}_i,\theta)$

Iterative Algorithm for Maximizing G

$\theta(1)$ = random, $Q(1)$ = random [not used]

$$Q(t+1) = \text{argmax}_Q \, G(Q,\theta(t)); \; Q_{iy}(t+1) = \Pr(y|\mathbf{x}_i, \theta(t))$$ **E**xpectation step

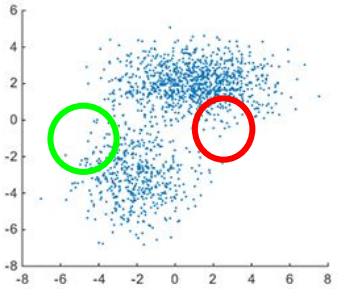
$$\theta(t+1) = \text{argmax}_\theta \, G(Q(t+1),\theta)$$ **M**aximization step

Note: $L(\theta(t+1)) \geq L(\theta(t))$.

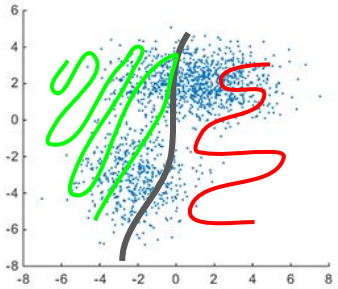
Why? $L(\theta(t+1)) = G(Q(t+2),\theta(t+1)) \geq G(Q(t+1),\theta(t+1)) \geq G(Q(t+1),\theta(t)) = L(\theta(t))$.

Run until improvements level off.

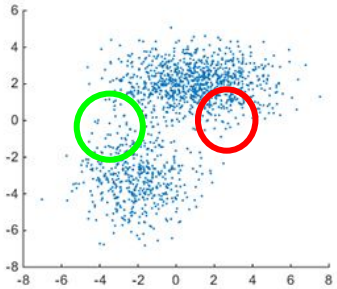
Visualization



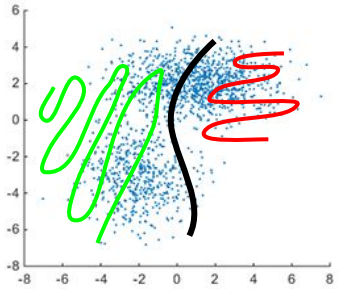
$\theta(1)$



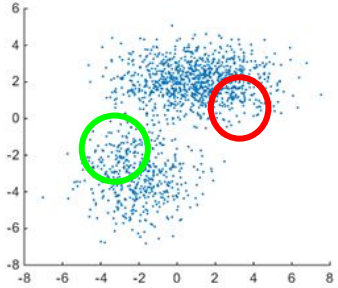
$Q(2)$



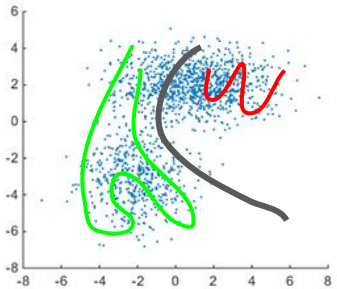
$\theta(2)$



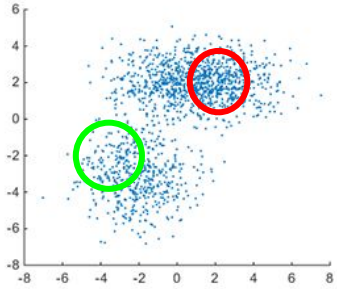
$Q(3)$



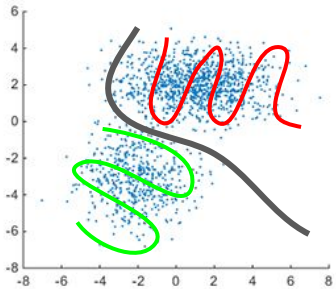
$\theta(3)$



$Q(4)$



$\theta(4)$



$Q(5)$

EM For K Gaussians

Expectation step:

$$Q_{iy} = \Pr(y|\mathbf{x}_i, \theta) \propto \Pr(\mathbf{x}_i|y, \theta) \Pr(y|\theta) = \Pr(\mathbf{x}_i|y, \mu_y) \Pr(y|c_y)$$
$$= 1/(2\pi)^{d/2} \exp(-½ (\mathbf{x}_i-\mu_y)^T (\mathbf{x}_i-\mu_y)) c_y$$

Maximization step: (Fractional counts.)

$$\mu_y = \sum_{i \in \{1,...,m\}} Q_{iy} \mathbf{x}_i / \sum_{i \in \{1,...,m\}} Q_{iy}$$

$$c_y = \sum_{i \in \{1,...,m\}} Q_{iy} / \sum_{y' \in \{1,...,k\}} \sum_{i \in \{1,...,m\}} Q_{iy'}$$

If were Q_{iy} binary, that's the mean of $\mathbf{x}_i$ and a count of the ys, just like Naive Bayes!

Connection to K-means

It’s kinda like K means, isn’t it?

- Start off randomly.
- We have a representation for each cluster.
- We alternate: (1) reassigning points to their natural clusters,
- and (2) redefining the clusters based on the points in them.
- Each iteration moves us in the right direction.
- It need not reach the optimum.

Some people call K Gaussians “soft K means”.

Hidden Music Maker

What can you discern from these random notes?

Hidden Markov Models

$\mathcal{X} = \mathbb{R}^d$. $\mathbf{x} \in \mathcal{X}$ generated by: Mixtures of Gaussians

1. Choose y with probability c_y for $y \in \{1,...,k\}$.
2. Choose $\mathbf{x}$ from the normal distribution associated with y:

$$\Pr(\mathbf{x}|y) = 1/(2\pi)^{d/2} \exp(-\tfrac{1}{2} (\mathbf{x}-\mu_y)^{\mathrm{T}} (\mathbf{x}-\mu_y)).$$

3. Go to 1.

$\mathcal{X} = \mathbb{R}^d$. $\mathbf{x} \in \mathcal{X}$ generated by: Hidden Markov Model

(non iid!)

1. Choose y with probability c_y for $i \in \{1,...,k\}$.
2. Choose $\mathbf{x}$ from the normal distribution associated with y:

$$\Pr(\mathbf{x}|y) = 1/(2\pi)^{d/2} \exp(-\tfrac{1}{2} (\mathbf{x}-\mu_y)^{\mathrm{T}} (\mathbf{x}-\mu_y)).$$

3. Choose y' with probability $T_{y\,y'}$ for $y' \in \{1,...,k\}$.
4. Set $y = y'$. Go to 2.

Applications of HMMs

Language processing: Part of speech tagging, speech recognition, author id

Vision: Activity recognition, person recognition via gait

Planning: Plan recognition, learning POMDPs

More: Musical score following, bioinformatics

EM for HMM

E-step: Given parameters (c_y, $T_{y\,y''}$, μ_y) and a sequence of observations, compute the probability of each cluster y at each point in the sequence.

Harder problem than before because the right guess for the cluster at time t depends on the guess for the time points around it.

Elegant dynamic programming algorithm called “forward-backward” algorithm solves this problem.

M-step: Still quite simple. Given the probability of each cluster y at each point in the sequence, estimate the parameters (c_y, $T_{y\,y''}$, μ_y) using MLE on the (fractional) counts.

Part of Speech Tagging

the tall girl sees a dog toy

DET ADJ NOUN VERB DET NOUN NOUN

Part of speech = hidden state. Can compute the likelihood of the sentence along with any particular tagging (or summed over all possible).

	DET	NOUN	VERB	ADJ
DET	0.0	0.6	0.0	0.4
NOUN	0.0	0.3	0.7	0.0
VERB	0.4	0.5	0.0	0.1
ADJ	0.0	0.8	0.0	0.2

DET	0.6 the	0.4 a	
NOUN	0.2 dog	0.3 girl	0.5 toy
VERB	0.1 sees	0.5 walks	0.4 sells
ADJ	0.2 tall	0.2 sullen	0.6 happy

Ambiguity and Loss

Gets interesting when things are ambiguous.

fly fly fly bloonoezbloonoezbloonoezeetee

VERB ADJ NOUN

Can compute the likelihood of the sentence along with any particular tagging (or summed over all possible taggings).

Can also compute the likelihood of each tag in each position.

What should we return?

What’s better?

A. Most likely sequence of states?
B. Sequence of most likely states?
C. Depends on your loss function.
D. You kind of gave that away, didn’t you?
E. None of the above.

Loss for Sequences

HMM:

class green

	end	NOUN	VERB	ADJ
start	0.0	0.6	0.0	0.4
NOUN	0.0	0.3	0.7	0.0
VERB	0.4	0.5	0.0	0.1
ADJ	0.0	0.8	0.0	0.2

DET	0.6 the	0.4 a		
NOUN	0.2 dog	0.3 girl	0.1 toy	0.5 ball
VERB	0.1 sees	0.4 walks	0.2 sells	0.4 chases
ADJ	0.2 tall	0.2 sullen	0.2 happy	

Viterbi

Most likely sequence of states ≠ sequence of most likely states.

Sometimes nice to find a single, self-consistent sequence.

Citation	Field
Viterbi (1967)	information theory
Vintsyuk (1968)	speech processing
Needleman & Wunsch (1970)	molecular biology
Sakoe & Chiba (1971)	speech processing
Sankoff (1972)	molecular biology
Reichert et al. (1973)	molecular biology
Wagner & Fischer (1974)	computer science

(from Speech and Language Processing. Jurafsky & Martin, https://web.stanford.edu/~jurafsky/slp3/9.pdf)

Representation

Decision trees: expressive Boolean attributes, continuous attributes

Naive Bayes: Generative, highly conditionally independent process (with latent information)

Nearest neighbor

SVM: linear separator, combines with kernel trick

Linear regression

Neural networks: DAGs of simple non-linear units

Loss

0-1 loss: right vs. wrong

Mean squared loss: MSE (mean squared error)

Likelihood.

Margin (slack version, too)

Regularization: penalizing “complex” rules. (Sum of squared weights. Degree of polynomial. Number of nodes. Number of latent clusters.).

Absolute loss.

Optimization

SVD. Finds the minimum squared reconstruction error. Matrix inversion.

SGD. Stochastic gradient descent. Iterative algorithm. Local optimum. Relative efficient. Derivatives. (Backprop.)

Solve for local minimum using derivatives.

Brute force: Go through all the choices.

Linear program. Quadratic program.

EM. Two-way iterative improvement.

Maximum likelihood.