CPSC 540: Machine Learning

Markov Chains

Mark Schmidt

University of British Columbia

Winter 2019

Last Time: PCA vs. Factor Analysis

- We discussed probabilistic PCA where we assume

$$x^i \mid z^i \sim \mathcal{N}(W^T z^i, \sigma^2 I), \quad z^i \sim \mathcal{N}(0, I),$$

  and we obtain PCA as $\sigma \to 0$.

- We discussed factor analysis (replaces $\sigma^2 I$ with diagonal D.

- Differences of FA with PCA:
  - FA is Not affected by scaling individual features.
    - FA doesn't chase large-noise features that are uncorrelated with other features.
  - But unlike PCA, it's affected by rotation of the data $(XQ$ vs. $X)$.
  - No nice "SVD" approach for FA, you can get different local optima.
  - In practice, not a big difference.

Independent Component Analysis (ICA)

- Factor analysis has found an enormous number of applications.
  - People really want to find the "factors" that make up their data.

- But even in ideal settings factor analysis can't uniquely identify the true factors.
  - We can rotate W and obtain the same model.

- Independent component analysis (ICA) is a more recent approach.
  - Around 30 years old instead of > 100.
  - Under certain assumptions, it can identify factors.
  - Canonical applications: blind source separation, identifying causal direction.

- It's the only algorithm we didn't cover in 340 from the list of "The 10 Algorithms Machine Learning Engineers Need to Know".

- I put last year's material on probabilistic PCA, factor analysis, and ICA here:
  - `https://www.cs.ubc.ca/~schmidtm/Courses/540-W19/L17.5.pdf`

End of Part 2: Basic Density Estimation and Mixture Models

- We defined the problem of density estimation
  - Computing probability of new examples $\tilde{x}^i$.
- We discussed basic distributions for 1D-case:
  - Bernoulli, categorical, Gaussian.
- We discussed product of independent distributions:
  - Model each feature individually.
- We discussed multivariate Gaussian:
  - Joint Gaussian model of multiple variables.

End of Part 2: Basic Density Estimation and Mixture Models

- We discussed mixture models:
  - Write density as a convex combination of densities.
  - Examples include mixture of Gaussians and mixture of Bernoullis.
  - Can model multi-modal densities.
- Commonly-fit using expectation maximization.
  - Generic method for dealing with missing at random data.
  - Can be viewed as a "minimize upper bound" method.
- Kernel density estimation is a non-parametric mixture model.
  - Place on mixture component on each data point.
  - Nice for visualizing low-dimensional densities.

Outline

1. Markov Chains

2. [In]Homogeneous Markov Chains

Example: Vancouver Rain Data

- Consider density estimation on the "Vancouver Rain" dataset:

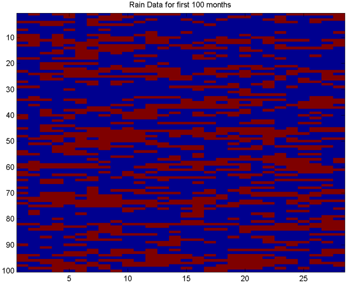
	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Day 8	Day 9	...
Month 1	0	0	0	1	1	0	0	1	1	
Month 2	1	0	0	0	0	0	1	0	0	
Month 3	1	1	1	1	1	1	1	1	1	
Month 4	1	1	1	1	0	0	1	1	1	
Month 5	0	0	0	0	1	1	0	0	0	
Month 6	0	1	1	0	0	0	0	1	1	

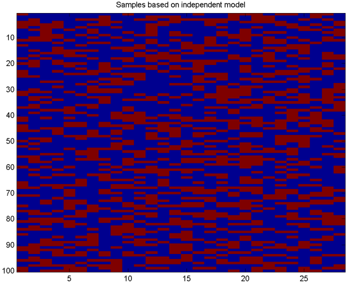
- Variable $x_j^i = 1$ if it rained on day j in month i.
  - Each row is a month, each column is a day of the month.
  - Data ranges from 1896-2004.

- The strongest signal in the data is the simple relationship:
  - If it rained yesterday, it's likely to rain today ($> 50\%$ chance of $(x_j^i == x_{j-1}^i)$).

Example: Vancouver Rain Data

- With independent Bernoullis, we get $p(x_j^i = \text{"rain"}) \approx 0.41$ (sadly).
  - Real data vs. product of Bernoullis model (red means "rain"):

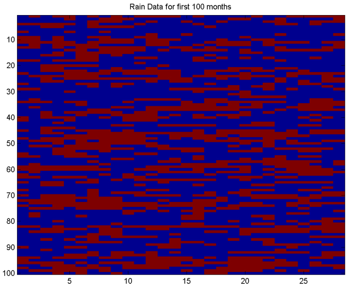


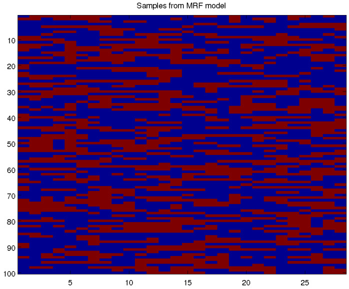


- Making days independent misses correlations.

Markov Chains

- A better density model for this data is a Markov chain.
  - Models $p(x_j^i \mid x_{j-1}^i)$: probability of rain today given yesterday's value.
    - Captures dependency between adjacent days.





- Mixture of Bernoullis can also model correlations, but it's inefficient:
  - Doesn't account for "position independence" of correlation.
  - Need clusters that correlate day 1 and 2, that correlate day 2 and 3, and so on.

Markov Chain Ingredients

- Markov chain ingredients:
  - State space:
    - Set of possible states (indexed by c) we can be in at time j ("rain" or "not rain").
  - Initial probabilities:
    - $p(x_1 = c)$: probability that we start in state c at time $j = 1$ (p("rain") on day 1).
  - Transition probabilities:
    - $p(x_j = c \mid x_{j-1} = c')$: probability that we move from state c' to state c at time j.
    - Probability that it rains today, given what happened yesterday.

- Notation alert: I'm going to start using "x_j" as short for "x_j^i" for a generic i.

- We're assuming a meaningful ordering of features.
  - We're modeling dependency of each feature on the previous feature.

Markov Chains

- By using the product rule, $p(a, b) = p(a)p(b \mid a)$, we can write any density as

$$\begin{aligned} p(x_1, x_2, \dots, x_d) &= p(x_1)p(x_2, x_3, \dots, x_d \mid x_1) \\ &= p(x_1)p(x_2 \mid x_1)p(x_3, x_4, \dots, x_d \mid x_1, x_2) \\ &= p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_2, x_1)p(x_4, x_5, \dots, x_d \mid x_1, x_2, x_3), \end{aligned}$$

  and so on until we get

$$p(x_1, x_2, \dots, x_d) = p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_2, x_1) \cdots p(x_d \mid x_{d-1}, x_{d-2}, \dots, x_1).$$

- This factorization of a density is called the chain rule of probability.
- But it leads to complicated conditionals:
  - For binary x_j, we need 2^d parameters for $p(x_d \mid x_1, x_2, \dots, x_{d-1})$ alone.

Markov Chains

- Markov chains simplify the distribution by assuming the Markov property:

$$p(x_j \mid x_{j-1}, x_{j-2}, \dots, x_1) = p(x_j \mid x_{j-1}),$$

that x_j is independent of the past given x_{j-1}.

  - To predict "rain", the only relevant past information is whether it rained yesterday.

- The probability for a sequence $x_1, x_2, \cdots, x_d$ in a Markov chain simplifies to

$$\begin{aligned} p(x_1, x_2, \dots, x_d) &= p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_2, x_1) \cdots p(x_d \mid x_{d-1}, x_{d-2}, \dots, x_1) \\ &= p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_2) \cdots p(x_d \mid x_{d-1}) \end{aligned}$$

- Another way to write the joint probability is

$$p(x_1, x_2, \dots, x_d) = \underbrace{p(x_1)}_{\text{initial prob.}} \prod_{j=2}^{d} \underbrace{p(x_j \mid x_{j-1})}_{\text{transition prob.}}.$$

Markov Chains

- Markov chains are ubiquitous in sequence/time-series models:

  9 Applications
  - 9.1 Physics
  - 9.2 Chemistry
  - 9.3 Testing
  - 9.4 Speech Recognition
  - 9.5 Information sciences
  - 9.6 Queueing theory
  - 9.7 Internet applications
  - 9.8 Statistics
  - 9.9 Economics and finance
  - 9.10 Social sciences
  - 9.11 Mathematical biology
  - 9.12 Genetics
  - 9.13 Games
  - 9.14 Music
  - 9.15 Baseball
  - 9.16 Markov text generators

Homogenous Markov Chains

- For rain data it makes sense to use a homogeneous Markov chain:
  - Transition probabilities $p(x_j \mid x_{j-1})$ are the same for all j.

- With discrete states, we could parameterize transition probabilities by

$$p(x_j = c \mid x_{j-1} = c') = \theta_{c,c'},$$

where $\theta_{c,c'} \geq 0$ and $\sum_{c=1}^{k} \theta_{c,c'} = 1$ (and we use the same $\theta_{c,c'}$ for all j).
  - So we have a categorical distribution over c values for each c' value.

- MLE for homogeneous Markov chain with discrete x_j is:

$$\theta_{c,c'} = \frac{(\text{number of transitions from } c' \text{ to } c)}{(\text{number of times we went from } c' \text{ to anything})},$$

so learning is just counting.

Parameter Tieing

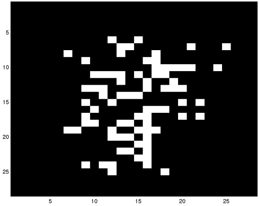
- Using same parameters $\theta_{c,c'}$ for different j is called parameter tieing.
  - "Making different parts of the model use the same parameters."

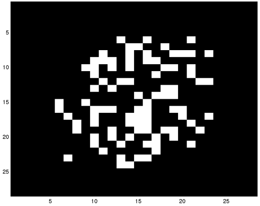
- Key advantages to parameter tieing:
  1. You have more data available to estimate each parameter.
     - Don't need to independently learn $p(x_j \mid x_{j-1})$ for days 3 and 24.
  2. You can have training examples of different sizes.
     - Same model can be used for any number of days.
     - We could even treat the data as one long Markov chain ($n = 1$).

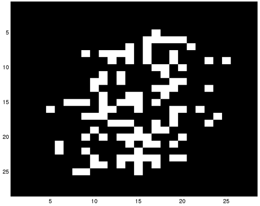
- We've seen parameter tieing before:
  - In 340 we discussed convolutional neural networks, which repeat same filters.
  - Throughout 340/540, we've assumed tied parameters across training examples.
    - That you use the same parameter for x^i and x^j.
    - Mixtures models relax this (same parameters only within cluster).

Density Estimation for MNIST Digits

- We've previously considered density estimation for MNIST images of digits.
- We saw that independent Bernoullis do terrible

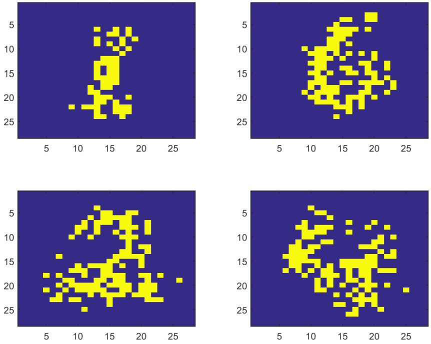






Density Estimation for MNIST Digits

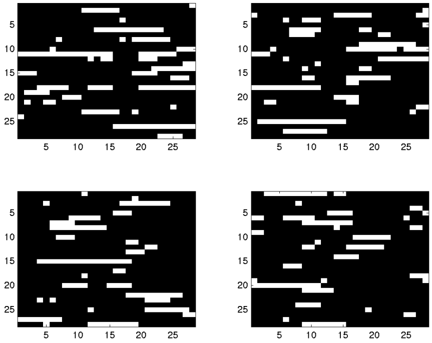
- We can do a bit better with mixture of 10 Bernoullis:



- The shape is looking better, but it's missing correlation between adjacent pixels.
  - Could we capture this with a Markov chain?

Density Estimation for MNIST Digits

- Samples from a homogeneous Markov chain (putting rows into one long vector):



- Captures correlations between adjacent pixels in the same row.
  - But misses long-range dependencies in row and dependencies between rows.
  - Also, "position independence" of homogeneity means it loses position information.

Inhomogeneous Markov Chains

- Markov chains could allow a different $p(x_j \mid x_{j-1})$ for each j.
- For discrete x_j we could use

$$p(x_j = c \mid x_{j=1} = c') = \theta^j_{c,c'}.$$

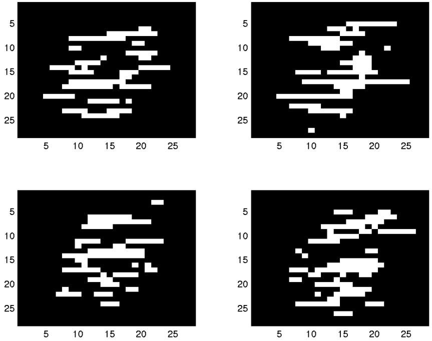
- MLE for discrete x_j values is given by

$$\theta^j_{c,c'} = \frac{(\text{number of transitions from } c' \text{ to } c \text{ starting at } (j-1))}{(\text{number of times we saw } c' \text{ at position } (j-1))},$$

- Such inhomogeneous Markov chains include independent models as special case:
  - We could set $p(x_j \mid x_{j-1}) = p(x_j)$.

Density Estimation for MNIST Digits

- Samples from an inhomogeneous Markov chain:



- We have correlations between adjacent pixels in rows and position information.
  - But isn't capturing long-range dependencies or dependency between rows.
  - Later we'll discuss graphical models which address this.
  - You could alternately consider a mixture of Markov chains.

Computation with Markov Chains

- Common things we do with Markov chains:
  1. Sampling: generate sequences that follow the probability.
  2. Marginalization: compute probability of being in state c at time j.
  3. Decoding: compute most likely sequence of states.
     - Decoding and marginalization will be important when we return to supervised learning.
  4. Conditioning: do any of the above, assuming $x_j = c$ for some j and c.
     - For example, "filling in" missing parts of the image.
  5. Stationary distribution: probability of being in state c as j goes to ∞.
     - Usually for homogeneous Markov chains.

Fun with Markov Chains

- Markov Chains “Explained Visually”: `http://setosa.io/ev/markov-chains`
- Snakes and Ladders: `http://datagenetics.com/blog/november12011/index.html`
- Candyland: `http://www.datagenetics.com/blog/december12011/index.html`
- Yahtzee: `http://www.datagenetics.com/blog/january42012/`
- Chess pieces returning home and K-pop vs. ska: `https://www.youtube.com/watch?v=63HHmjlh794`

Summary

- Markov chains model dependencies between adjacent features.
- Parameter tieing uses same parameters in different parts of a model.
  - Example of "homogeneous" Markov chain.
  - Allows models of different sizes and more data per parameter.
- Markov chain tasks:
  - Sampling, marginalization, decoding, conditioning, stationary distributions.
- Next time: the other "MC" in MCMC.

Scale Mixture Models

- Another weird mixture model is a scale mixture of Gaussians,

$$p(x^i) = \int_{\sigma^2} p(\sigma^2)\mathcal{N}(x^i \mid \mu, \sigma^2) d\sigma^2.$$

- Common choice for $p(\sigma^2)$ is a gamma distribution (which makes integral work):
  - Many distributions are special cases, like Laplace and student t.
- Leads to EM algorithms for fitting Laplace and student t.