CPSC 540: Machine Learning
Markov Chains

Mark Schmidt
University of British Columbia

Winter 2020
Example: Vancouver Rain Data

Consider density estimation on the “Vancouver Rain” dataset:

Variable $x^i_j = 1$ 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 signals in the data:
- It tends to rain more in the winter than the summer.
- If it rained yesterday, it’s likely to rain today ($> 50\%$ chance of $(x^i_j = x^i_{j-1})$).
Rain Data with Independent Bernoullis

- With independent Bernoullis, we get \( p(x^i_j = \text{"rain"}) \approx 0.41 \) (sadly).
- Samples from product of Bernoullis model (left) vs. real data (right):

- Making days independent misses seasons and misses correlations.
A better model is a mixture of Bernoullis:

- Samples from product of Bernoullis model (left) vs. mixture of 50 Bernoullis (right):

- Mixture of Bernoullis can learn that there are seasons (clusters).
- But mixture of Bernoullis can’t easily learn the between-day correlations.
Rain Data with Mixture of Bernoullis

- Visualizing the mean parameters of the mixture of 50 Bernoullis:

- Recall that mixture of Bernoullis assumes independence, given cluster.
- This makes it try to model between-day correlations in a weird way:
  - Uses clusters with rain for consecutive days, during different parts of month.

- So you would need a lot of clusters to model all between-day correlations.
  - Doesn’t account for “position independence” of the correlation.
  - Need cluster that correlate that day 1 and 2, that correlate day 2 and 3, and so on.
A better model for the between-day correlations is a Markov chain.

- Models $p(x^i_j \mid x^i_{j-1})$: probability of rain today given yesterday’s value.
- Captures dependency between adjacent days.

- It can perfectly capture the “position-independent” between-day correlation.
  - With only a few parameters and a closed-form MLE (no EM or non-convexity).
Markov Chain for Rain

- Markov chain ingredients and MLE for rain data:
  - **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 "\( x_i^j \)" for a generic \( i \).

- We’re assuming that the order of features is meaningful.
  - We’re modeling dependency of each feature on the previous feature.
Markov Chain Ingredients

- **Markov chain ingredients** and MLE for rain data:
  - **State space:**
    - At time $t$, we can be in the “rain” state or the “not rain” state.
  - **Initial probabilities:**
    
    | State   | Probability |
    |---------|-------------|
    | Rain    | 0.37        |
    | Not Rain| 0.63        |
  
  - **Transition probabilities:**
    
    | $c'$     | $c$  | $p(x_j = c \mid x_{j-1} = c')$ |
    |----------|------|---------------------------------|
    | Rain     | Rain | 0.65                            |
    | Rain     | Not Rain | 0.35                   |
    | Not Rain | Rain | 0.25                            |
    | Not Rain | Not Rain | 0.75                     |
  
  - Because of “sum to 1” constraints, there are **only 3 parameters** in this model.
Chain Rule of Probability

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

\[
p(x_1, x_2, \ldots, x_d) = p(x_1)p(x_2, x_3, \ldots, x_d \mid x_1) \\
= p(x_1)p(x_2 \mid x_1)p(x_3, x_4, \ldots, x_d \mid x_1, x_2) \\
= p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_2, x_1)p(x_4, x_5, \ldots, x_d \mid x_1, x_2, x_3),
\]

and so on until we get

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

- 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 \( p(x_d \mid x_1, x_2, \ldots, x_{d-1}) \) alone.
Markov Chains

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

\[ p(x_j \mid x_{j-1}, x_{j-2}, \ldots, x_1) = p(x_j \mid x_{j-1}), \]

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 \( x_1, x_2, \ldots, x_d \) in a Markov chain simplifies to

\[
p(x_1, x_2, \ldots, x_d) = p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_2, x_1) \cdots p(x_d \mid x_{d-1}, x_{d-2}, \ldots, x_1) \\
= p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_2) \cdots p(x_d \mid x_{d-1})
\]

- Another way to write the joint probability is

\[
p(x_1, x_2, \ldots, x_d) = p(x_1) \prod_{j=2}^{d} p(x_j \mid x_{j-1}).
\]

\[
\text{initial prob.} \quad \text{transition prob.}
\]
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 $p(x_\ldots \mid x_{\ldots-1})$ are the same for all $\ldots$.

- With discrete states, we could parameterize transition probabilities by
  
  $p(x_\ldots = c \mid x_{\ldots-1} = c') = \theta_{c,c}$

  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 $\ldots$).

  - So we have a categorical distribution over $c$ values for each $c'$ value.

- MLE for homogeneous Markov chain with discrete $x_\ldots$ 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 $p(x_j | x_{j-1})$ 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 $x^j$.
    - Can think of mixtures models as relaxing this (same parameters only within cluster).
Example: Modeling DNA Sequences

- A nice demo of independent vs. Markov (and HMMs) for DNA sequences:

- Independent model for elements of sequence:

![DNA structure image](https://www.tes.com/lessons/WE5E9RncBhieAQ/dna)
Example: Modeling DNA Sequences

- **Transition probabilities** in a Markov chain model for elements of sequence:

  (visualizing transition probabilities based on previous symbol):
Density Estimation for MNIST Digits

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

![Images of MNIST digits](image1.png)

- We saw that a mixture of Bernoullis does better:

![Images of MNIST digits](image2.png)

- 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 $j$.
  - This makes sense for digits data, but probably not for the rain data.

- For discrete $x_j$ we could use

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

- MLE for discrete $x_j$ values is given by

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

- 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.
Some common setups for fitting the parameters Markov chains:

1. We have one long sequence, and fit parameters of an homogeneous Markov chain.
   - Here, we just focus on the transition probabilities.

2. We have many sequences of different lengths, and fit a homogeneous chain.
   - And we can use it to model sequences of any length.

3. We have many sequences of same length, and fit an inhomogeneous Markov chain.
   - This allows “position-specific” effects.

4. We use domain knowledge to guess the initial and transition probabilities.
Inference in Markov Chains

Given a Markov chain model, these are the most common inference tasks:

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 $\infty$.
   - 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 tying** 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.