


Feature-based Discriminative Classifiers

Making features from text for discriminative NLP models

Christopher Manning

Christopher Manning




Classifiers

- A classifier is a function f which assigns an input datum d to one of $|C|$ classes, $c \in C$
- The classes might be:
 - {spam, notspam} for an email message
 - {politics, sports, finance, technology, arts, leisure, ...} for news
 - {we-are-coreferent, we-are-not-coreference} for a coreference candidate mention pair

2


Christopher Manning



Example problem

- Classify a capitalized proper noun as a class:
 - LOCATION, DRUG, PERSON
- For a data example d
 - taking Zantac
- We work by considering each class c for the word:
 - (LOCATION, taking Zantac,)
 - (DRUG, taking Zantac,)
 - (PERSON, taking Zantac,)
- and using features to score each candidate classification


Christopher Manning



Features for a classifier

- Features f are elementary pieces of evidence that link aspects of what we observe d with a category c that we want to predict
- A feature is a function with a bounded real value: $f: C \times D \rightarrow \mathbb{R}$
 - Common special case:
 - binary features $f: C \times D \rightarrow \{0, 1\}$

Christopher Manning



Example features

- $f_1(c, d) = [c = \text{LOCATION} \wedge w_1 = \text{"in"} \wedge \text{isCapitalized}(w)]$
- $f_2(c, d) = [c = \text{LOCATION} \wedge \text{hasAccentedLatinChar}(w)]$
- $f_3(c, d) = [c = \text{DRUG} \wedge \text{ends}(w, \text{"c"})]$


1.8 -0.6 0.3 PERSON

LOCATION LOCATION DRUG PERSON

in Arcadia in Québec taking Zantac saw Sue

- Models will assign to each feature a **weight**:
 - A positive weight votes that this configuration is likely correct
 - A negative weight votes that this configuration is likely incorrect

Christopher Manning




Features

- Very commonly, a feature specifies
 - an indicator function – a yes/no boolean matching function – of properties of the input *and*
 - a particular class

$$f_i(c, d) = [\Phi(d) \wedge c = c_i] \quad \text{[Value is 0 or 1]}$$

- Each feature picks out a data subset and suggests a label for it



Feature-Based Models

- The decision about a data point is based only on the **features** active at that point.

Data
BUSINESS: Stocks
hit a yearly low ...


Label: BUSINESS
Features
 $\{..., \text{hit}, a, \text{yearly}, \text{low}, \dots\}$

Data
... to restructure
bank: MONEY debt.

Label: MONEY
Features
 $\{..., w_1 = \text{restructure}, w_{n-1} = \text{debt}, \text{Leng} = 12, \dots\}$

Text
Categorization

Word-Sense
Disambiguation



Feature-Based Linear Classifiers

- Linear classifiers at classification time:
 - Linear function from feature sets $\{f_i\}$ to classes $\{c\}$.
 - Assign a weight λ_i to each feature f_i .
 - We consider each class for an observed datum d
 - For a pair (c, d) , features vote with their weights:
 - $\text{vote}(c) = \sum \lambda_i f_i(c, d)$

PERSON

in Québec

LOCATION


in Québec

DRUG

in Québec

- Choose the class c which maximizes $\sum \lambda_i f_i(c, d)$

Christopher Manning



Feature-Based Linear Classifiers

- Linear classifiers at classification time:
 - Linear function from feature sets $\{f_i\}$ to classes $\{c\}$.
 - Assign a weight λ_i to each feature f_i .
 - We consider each class for an observed datum d
 - For a pair (c, d) , features vote with their weights:
 - $\text{vote}(c) = \sum \lambda_i f_i(c, d)$


PERSON
in Québec

1.8 LOCATION
in Québec -0.6

0.3 DRUG
in Québec

- Choose the class c which maximizes $\sum \lambda_i f_i(c, d)$ = LOCATION

Christopher Manning

 **Feature-Based Linear Classifiers**

There are many ways to choose weights for features

- Perceptron: find a currently misclassified example, and nudge weights in the direction that corrects classification
- Margin-based methods (Support Vector Machines)
- Maximum entropy models (“softmax regression”; roughly logistic regression), which we will look at next

The word cloud features the following terms:

- sentences
- among
- probability
- grammar
- model
- algorithm
- words
- language
- system
- rules
- information
- feature
- different
- given
- number
- planning
- processes
- variation
- models
- sequence
- represented
- discrete
- known
- used
- not
- task
- rule
- learning
- using
- very
- representing
- separate
- new
- compositional
- rules
- in
- to
- of
- a
- the
- and
- on
- with
- from
- by
- at
- into
- out
- up
- down
- off
- over
- under
- through
- across
- against
- between
- within
- without
- above
- below
- beside
- near
- far
- close
- open
- empty
- full
- half
- quarter
- third
- fourth
- fifth
- sixth
- seventh
- eighth
- ninth
- tenth
- eleventh
- twelfth
- thirteenth
- fourteenth
- fifteenth
- sixteenth
- seventeenth
- eighteenth
- nineteenth
- twentieth
- twenty-first
- twenty-second
- twenty-third
- twenty-fourth
- twenty-fifth
- twenty-sixth
- twenty-seventh
- twenty-eighth
- twenty-ninth
- thirtieth
- thirty-first
- thirty-second
- thirty-third
- thirty-fourth
- thirty-fifth
- thirty-sixth
- thirty-seventh
- thirty-eighth
- thirty-ninth
- fortieth
- forty-first
- forty-second
- forty-third
- forty-fourth
- forty-fifth
- forty-sixth
- forty-seventh
- forty-eighth
- forty-ninth
- fiftieth
- fifty-first
- fifty-second
- fifty-third
- fifty-fourth
- fifty-fifth
- fifty-sixth
- fifty-seventh
- fifty-eighth
- fifty-ninth
- sixtieth
- sixty-first
- sixty-second
- sixty-third
- sixty-fourth
- sixty-fifth
- sixty-sixth
- sixty-seventh
- sixty-eighth
- sixty-ninth
- seventieth
- seventy-first
- seventy-second
- seventy-third
- seventy-fourth
- seventy-fifth
- seventy-sixth
- seventy-seventh
- seventy-eighth
- seventy-ninth
- eightieth
- eighty-first
- eighty-second
- eighty-third
- eighty-fourth
- eighty-fifth
- eighty-sixth
- eighty-seventh
- eighty-eighth
- eighty-ninth
- ninetieth
- ninety-first
- ninety-second
- ninety-third
- ninety-fourth
- ninety-fifth
- ninety-sixth
- ninety-seventh
- ninety-eighth
- ninety-ninth
- hundredth
- hundred-first
- hundred-second
- hundred-third
- hundred-fourth
- hundred-fifth
- hundred-sixth
- hundred-seventh
- hundred-eighth
- hundred-ninth
- hundred-tenth
- hundred-eleventh
- hundred-twelfth
- hundred-thirteenth
- hundred-fourteenth
- hundred-fifteenth
- hundred-sixteenth
- hundred-seventeenth
- hundred-eighteenth
- hundred-nineteenth
- hundred-twentieth
- hundred-twenty-first
- hundred-twenty-second
- hundred-twenty-third
- hundred-twenty-fourth
- hundred-twenty-fifth
- hundred-twenty-sixth
- hundred-twenty-seventh
- hundred-twenty-eighth
- hundred-twenty-ninth
- hundred-thirtieth
- hundred-thirty-first
- hundred-thirty-second
- hundred-thirty-third
- hundred-thirty-fourth
- hundred-thirty-fifth
- hundred-thirty-sixth
- hundred-thirty-seventh
- hundred-thirty-eighth
- hundred-thirty-ninth
- hundred-fortieth
- hundred-forty-first
- hundred-forty-second
- hundred-forty-third
- hundred-forty-fourth
- hundred-forty-fifth
- hundred-forty-sixth
- hundred-forty-seventh
- hundred-forty-eighth
- hundred-forty-ninth
- hundred-fiftieth
- hundred-fifty-first
- hundred-fifty-second
- hundred-fifty-third
- hundred-fifty-fourth
- hundred-fifty-fifth
- hundred-fifty-sixth
- hundred-fifty-seventh
- hundred-fifty-eighth
- hundred-fifty-ninth
- hundred-sixtieth
- hundred-sixty-first
- hundred-sixty-second
- hundred-sixty-third
- hundred-sixty-fourth
- hundred-sixty-fifth
- hundred-sixty-sixth
- hundred-sixty-seventh
- hundred-sixty-eighth
- hundred-sixty-ninth
- hundred-seventieth
- hundred-seventy-first
- hundred-seventy-second
- hundred-seventy-third
- hundred-seventy-fourth
- hundred-seventy-fifth
- hundred-seventy-sixth
- hundred-seventy-seventh
- hundred-seventy-eighth
- hundred-seventy-ninth
- hundred-eightieth
- hundred-eighty-first
- hundred-eighty-second
- hundred-eighty-third
- hundred-eighty-fourth
- hundred-eighty-fifth
- hundred-eighty-sixth
- hundred-eighty-seventh
- hundred-eighty-eighth
- hundred-eighty-ninth
- hundred-ninetieth
- hundred-ninety-first
- hundred-ninety-second
- hundred-ninety-third
- hundred-ninety-fourth
- hundred-ninety-fifth
- hundred-ninety-sixth
- hundred-ninety-seventh
- hundred-ninety-eighth
- hundred-ninety-ninth
- one thousand
- one thousand one
- one thousand two
- one thousand three
- one thousand four
- one thousand five
- one thousand six
- one thousand seven
- one thousand eight
- one thousand nine
- one thousand ten
- one thousand eleven
- one thousand twelve
- one thousand thirteen
- one thousand fourteen
- one thousand fifteen
- one thousand sixteen
- one thousand seventeen
- one thousand eighteen
- one thousand nineteen
- one thousand twenty
- one thousand twenty-one
- one thousand twenty-two
- one thousand twenty-three
- one thousand twenty-four
- one thousand twenty-five
- one thousand twenty-six
- one thousand twenty-seven
- one thousand twenty-eight
- one thousand twenty-nine
- one thousand thirty
- one thousand thirty-one
- one thousand thirty-two
- one thousand thirty-three
- one thousand thirty-four
- one thousand thirty-five
- one thousand thirty-six
- one thousand thirty-seven
- one thousand thirty-eight
- one thousand thirty-nine
- one thousand forty
- one thousand forty-one
- one thousand forty-two
- one thousand forty-three
- one thousand forty-four
- one thousand forty-five
- one thousand forty-six
- one thousand forty-seven
- one thousand forty-eight
- one thousand forty-nine
- one thousand fifty
- one thousand fifty-one
- one thousand fifty-two
- one thousand fifty-three
- one thousand fifty-four
- one thousand fifty-five
- one thousand fifty-six
- one thousand fifty-seven
- one thousand fifty-eight
- one thousand fifty-nine
- one thousand sixty
- one thousand sixty-one
- one thousand sixty-two
- one thousand sixty-three
- one thousand sixty-four
- one thousand sixty-five
- one thousand sixty-six
- one thousand sixty-seven
- one thousand sixty-eight
- one thousand sixty-nine
- one thousand seventy
- one thousand seventy-one
- one thousand seventy-two
- one thousand seventy-three
- one thousand seventy-four
- one thousand seventy-five
- one thousand seventy-six
- one thousand seventy-seven
- one thousand seventy-eight
- one thousand seventy-nine
- one thousand eighty
- one thousand eighty-one
- one thousand eighty-two
- one thousand eighty-three
- one thousand eighty-four
- one thousand eighty-five
- one thousand eighty-six
- one thousand eighty-seven
- one thousand eighty-eight
- one thousand eighty-nine
- one thousand ninety
- one thousand ninety-one
- one thousand ninety-two
- one thousand ninety-three
- one thousand ninety-four
- one thousand ninety-five
- one thousand ninety-six
- one thousand ninety-seven
- one thousand ninety-eight
- one thousand ninety-nine
- two thousand
- two thousand one
- two thousand two
- two thousand three
- two thousand four
- two thousand five
- two thousand six
- two thousand seven
- two thousand eight
- two thousand nine
- two thousand ten
- two thousand eleven
- two thousand twelve
- two thousand thirteen
- two thousand fourteen
- two thousand fifteen
- two thousand sixteen
- two thousand seventeen
- two thousand eighteen
- two thousand nineteen
- two thousand twenty
- two thousand twenty-one
- two thousand twenty-two
- two thousand twenty-three
- two thousand twenty-four
- two thousand twenty-five
- two thousand twenty-six
- two thousand twenty-seven
- two thousand twenty-eight
- two thousand twenty-nine
- two thousand thirty
- two thousand thirty-one
- two thousand thirty-two
- two thousand thirty-three
- two thousand thirty-four
- two thousand thirty-five
- two thousand thirty-six
- two thousand thirty-seven
- two thousand thirty-eight
- two thousand thirty-nine
- two thousand forty
- two thousand forty-one
- two thousand forty-two
- two thousand forty-three
- two thousand forty-four
- two thousand forty-five
- two thousand forty-six
- two thousand forty-seven
- two thousand forty-eight
- two thousand forty-nine
- two thousand fifty
- two thousand fifty-one
- two thousand fifty-two
- two thousand fifty-three

[illegible]



Feature-Based Linear Classifiers

- Linear classifiers are a linear function from feature sets $\{f_i\}$ to classes $\{c\}$
- At test time, we consider each class c for a datum d
 - We generate a feature set $\{f_i\}$ for an observed datum-class pair (c, d)
 - Each feature f_i has a weight λ_i
 - We then score each possible class assignment: $\text{vote}(c) = \sum \lambda_i f_i(c, d)$
 - We choose the class c which maximizes $\sum \lambda_i f_i(c, d)$
- At training time we have observed (c, d) pairs from labeled examples
 - We generate sets of features $\{f_i(c, d)\}$ for them
 - We use information about what features occur and don't occur to set a weight λ_i for each feature



Example features

- $f_1(c, d) = [c = \text{LOCATION} \wedge w_{-1} = \text{"in"} \wedge \text{isCapitalized}(w)]$
- $f_2(c, d) = [c = \text{LOCATION} \wedge \text{hasAccentedLatinChar}(w)]$
- $f_3(c, d) = [c = \text{DRUG} \wedge \text{ends}(w, \text{"c"})]$

1.8 LOCATION -0.6 LOCATION 0.3 DRUG PERSON
in Arcadia in Québec taking Zantac saw Sue



Feature-Based Linear Classifiers

- Maxent (softmax, multiclass logistic, exponential, conditional log-linear, Gibbs) models:
 - Make a probabilistic model from the linear combination $\sum \lambda_i f_i(c, d)$
- $$P(c | d, \lambda) = \frac{\exp \sum \lambda_i f_i(c, d)}{\sum_c \exp \sum \lambda_i f_i(c, d)}$$
- ← Makes votes positive
← Normalizes votes
- $P(\text{LOCATION} | \text{in Québec}) = e^{1.8} e^{-0.6} / (e^{1.8} e^{-0.6} + e^{0.3} + e^0) = 0.586$
 - $P(\text{DRUG} | \text{in Québec}) = e^{0.3} / (e^{1.8} e^{-0.6} + e^{0.3} + e^0) = 0.238$
 - $P(\text{PERSON} | \text{in Québec}) = e^0 / (e^{1.8} e^{-0.6} + e^{0.3} + e^0) = 0.176$
- The **weights** are the **parameters** of the probability model, combined via a "soft max" function



Feature-Based Linear Classifiers

- Maxent models:
 - Given this model form, we choose parameters $\{\lambda_i\}$ that **maximize the conditional likelihood** of the data according to this model (as discussed later)
 - We construct not only classifications, but probability distributions over classifications.
 - There are other (good!) ways of discriminating classes – SVMs, boosting, even perceptrons – but these methods are not as trivial to interpret as distributions over classes.



Feature Expectations

- We will crucially make use of two **expectations**
 - actual or predicted counts of a feature firing:
- Empirical count (expectation) of a feature:

$$\text{empirical } E(f_i) = \sum_{(c, d) \in \text{Observed}(C, D)} f_i(c, d)$$
- Model expectation of a feature:

$$E(f_i) = \sum_{(c, d) \in C \times D} P(c, d) f_i(c, d)$$



Building a Maxent Model

- We define features (indicator functions) over data points
 - Features represent sets of data points which are distinctive enough to deserve model parameters.
 - Words, but also "word contains number", "word ends with ing", POS, syntactic structure, relation between two phrases, etc.
- We might simply encode each Φ feature as a unique String
 - A datum will give rise to a set of Strings: the active Φ features
 - Each feature $f_i(c, d) = [\Phi(d) \wedge c = c_i]$ gets a real number weight
- We concentrate on Φ features but the math uses i indices of f_i



Building a Maxent Model

- Features are normally added in big batches via feature templates
 - E.g., one feature template adds $\forall i,j$ observed: $\text{lastWord}=w_i \wedge c=c_j$
 - Another is: $\text{nextWord}=w_i \wedge c=c_j$. Each may add tens of thousands of features
- A model may be specified by the set of feature templates used
- Features are often added during model development to target errors
 - Often, the easiest thing to think of are features that mark bad combinations



Maxent Models and Discriminative Estimation

Generative vs. Discriminative models

Christopher Manning



Introduction

- So far we've mainly looked at "generative models"
 - Language models, IBM alignment models, PCFGs
- But there is much use of conditional or discriminative models in NLP, Speech, IR, and ML generally
- Because:
 - They give high accuracy performance
 - They make it easy to incorporate lots of linguistically important features
 - They allow automatic building of language independent, retargetable NLP modules



Joint vs. Conditional Models

- We have some data $\{(d, c)\}$ of paired observations d and hidden classes c .
- **Joint (generative) models** place probabilities over both observed data and the hidden stuff
 - They generate the observed data from the hidden stuff
 - All the classic StatNLP models:
 - n -gram models, Naive Bayes classifiers, hidden Markov models, probabilistic context-free grammars, IBM machine translation alignment models

$P(c, d)$



Joint vs. Conditional Models


- **Discriminative (conditional) models** take the data as given, and put a probability/score over hidden structure given the data:
 - Logistic regression, conditional loglinear or maximum entropy models, conditional random fields
 - Also, SVMs, (averaged) perceptron, etc. are discriminative classifiers (but not directly probabilistic)

$P(d)$



Conditional vs. Joint Likelihood

- A **joint** model gives probabilities $P(d, c) = P(c)P(d|c)$ and tries to maximize this joint likelihood.
 - It ends up trivial to choose weights: just count! (relative frequencies)
- A **conditional** model gives probabilities $P(c|d)$. It takes the data as given and models only the conditional probability of the class.
 - We seek to maximize conditional likelihood.
 - Harder to do (as we'll see...)
 - More closely related to classification error.



Christopher Manning

Conditional models work well:


Word Sense Disambiguation

Training Set	
Objective	Accuracy
Joint Like.	86.8
Cond. Like.	98.5

Test Set	
Objective	Accuracy
Joint Like.	73.6
Cond. Like.	76.1

- Even with exactly the same features, changing from joint to conditional estimation increases performance
- That is, we use the same smoothing, and the same word-class features, we just change the numbers (parameters)

(Klein and Manning 2002, using Senseval-1 Data)



Christopher Manning

PCFGs Maximize Joint, not Conditional Likelihood

VP

V

eat

46

VP

V NP PP

eat rice with chopsticks

6

VP


V NP PP

eat rice with chopsticks

2

1. What parse for *eat rice with chopsticks*?
2. How can you get the other parse?

Based on an example by Mark Johnson




Christopher Manning

Exponential Model Likelihood

- Maximum (Conditional) Likelihood Models :
 - Given a model form, we choose values of parameters λ_i to maximize the (conditional) likelihood of the data.
- For any given feature weights, we can calculate:
 - Data conditional likelihood
 - Derivative of the likelihood wrt each feature weight

$$\log P(C|D, \lambda) = \sum_{(c,d) \in \mathcal{R}(D)} \log P(c|d, \lambda) = \sum_{(c,d) \in \mathcal{R}(D)} \log \frac{\exp \sum \lambda_i f_i(c, d)}{\sum_{c'} \exp \sum \lambda_i f_i(c', d)}$$



Christopher Manning

The Likelihood Value


- The (log) conditional likelihood of iid* data (C, D) according to a maxent model is a function of the data and the parameters λ :

$$\log P(C|D, \lambda) = \log \prod_{(c,d) \in (C,D)} P(c|d, \lambda) = \sum_{(c,d) \in (C,D)} \log P(c|d, \lambda)$$
- If there aren't many values of c , it's easy to calculate:

$$\log P(C|D, \lambda) = \sum_{(c,d) \in (C,D)} \frac{\exp \sum_i \lambda_i f_i(c, d)}{\sum_c \exp \sum_i \lambda_i f_i(c', d)}$$

*A fancy statistics term meaning "independent and identically distributed". You normally need to assume this for anything formal to be derivable, even though it's never quite true in practice.

Christopher Manning



The Likelihood Value

- We can separate this into two components:

$$\log P(C \mid D, \lambda) = \sum_{(c,d) \in \mathcal{C}(D)} \log \exp \sum_i \lambda_i f_i(c,d) - \sum_{(c,d) \in \mathcal{C}(C,D)} \log \sum_{c'} \exp \sum_i \lambda_i f_i(c',d)$$
$$\log P(C \mid D, \lambda) = N(\lambda) - M(\lambda)$$

- We can maximize it by finding where the derivative is 0
- The derivative is the difference between the derivatives of each component



The Derivative I: Numerator

$$\begin{aligned}\frac{\partial N(\lambda)}{\partial \lambda_i} &= \frac{\partial \sum_{(c,d) \in \mathcal{C}(D)} \log \exp \sum_i \lambda_i f_i(c,d)}{\partial \lambda_i} = \frac{\partial \sum_{(c,d) \in \mathcal{C}(D)} \sum_i \lambda_i f_i(c,d)}{\partial \lambda_i} \\ &= \sum_{(c,d) \in \mathcal{C}(D)} \frac{\partial \sum_i \lambda_i f_i(c,d)}{\partial \lambda_i} \\ &= \sum_{(c,d) \in \mathcal{C}(D)} f_i(c,d)\end{aligned}$$

Derivative of the numerator is: the empirical count(f_i, c)



The Derivative II: Denominator

$$\begin{aligned}\frac{\partial M(\lambda)}{\partial \lambda_i} &= \frac{\partial \sum_{(c,d) \in \mathcal{C}(D)} \log \exp \sum_i \lambda_i f_i(c',d)}{\partial \lambda_i} \\ &= \sum_{(c,d) \in \mathcal{C}(D)} \frac{1}{\sum_{c'} \exp \sum_i \lambda_i f_i(c',d)} \frac{\partial \exp \sum_i \lambda_i f_i(c',d)}{\partial \lambda_i} \\ &= \sum_{(c,d) \in \mathcal{C}(D)} \frac{1}{\sum_{c'} \exp \sum_i \lambda_i f_i(c',d)} \sum_{c'} \frac{\exp \sum_i \lambda_i f_i(c',d)}{1} \frac{\partial \sum_i \lambda_i f_i(c',d)}{\partial \lambda_i} \\ &= \sum_{(c,d) \in \mathcal{C}(D)} \sum_{c'} \frac{\exp \sum_i \lambda_i f_i(c',d)}{\sum_{c'} \exp \sum_i \lambda_i f_i(c',d)} \frac{\partial \sum_i \lambda_i f_i(c',d)}{\partial \lambda_i} \\ &= \sum_{(c,d) \in \mathcal{C}(D)} \sum_{c'} P(c' | d, \lambda) f_i(c',d) = \text{predicted count}(f_i, \lambda)\end{aligned}$$



The Derivative III

$$\frac{\partial \log P(C | D, \lambda)}{\partial \lambda_i} = \text{actual count}(f_i, C) - \text{predicted count}(f_i, \lambda)$$

- The optimum parameters are the ones for which each feature's **predicted expectation** equals its **empirical expectation**. The optimum distribution is:
 - Always unique (but parameters may not be unique)
 - Always exists (if feature counts are from actual data).
- These models are also called maximum entropy models because we find the model having maximum entropy and satisfying the constraints: $E_p(f_j) = E_{\bar{p}}(f_j), \forall j$



Finding the optimal parameters

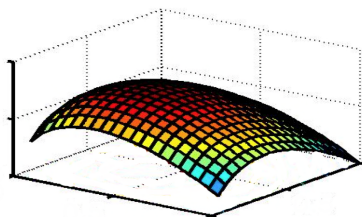
- We want to choose parameters $\lambda_1, \lambda_2, \lambda_3, \dots$ that maximize the conditional log-likelihood of the training data

$$CLogLik(D) = \sum_{i=1}^n \log P(c_i | d_i)$$

- To be able to do that, we've worked out how to calculate the function value and its partial derivatives (its gradient)



A likelihood surface



Finding the optimal parameters

- Use your favorite numerical optimization package....
 - Commonly (and in our code), you **minimize** the negative of $CLogLik$
- 1. Gradient descent (GD); Stochastic gradient descent (SGD)
- 2. Iterative proportional fitting methods: Generalized Iterative Scaling (GIS) and Improved Iterative Scaling (IIS)
- 3. Conjugate gradient (CG), perhaps with preconditioning
- 4. Quasi-Newton methods – limited memory variable metric (LMVM) methods, in particular, L-BFGS