

Document Labeling

(Mining the Web)

Soumen Chakrabarti

September 30, 2019

Document classification

- ▶ Applications: email routing, spam detection, Web directory maintenance, author identification, ...
- ▶ Document denoted $x \in X$, class label is a discrete y — could be $+1/-1$, topics, etc.
- ▶ Model joint distribution of X and Y , use Bayes rule
- ▶ Model conditional distribution of $Y|x$
- ▶ Discriminative classification: $\arg \max_y \beta^\top \phi(x, y)$
- ▶ For special case $y \in \{-1, 1\}$, can implement $\arg \max_y \beta^\top \phi(x, y)$ as $\text{sign}(\beta^\top \phi(x))$
- ▶ $\phi(x)$ or $\phi(x, y)$ is a **feature vector** (to be described)

Bayesian classification

- ▶ Estimate a corpus model for each class, $\Pr(x|y)$
- ▶ E.g., multinomial, Dirichlet, etc.
- ▶ Prior probability $\Pr(y)$ of class y is the fraction of training documents seen in class y
- ▶ Given test document x , $\Pr(y|x) \propto \Pr(y) \Pr(x|y)$
- ▶ Out-of-vocabulary words and the need for smoothing
- ▶ $\Pr(x|y)$ is very inaccurate
- ▶ Decision surface may not be very bad!

Conditional probabilistic classification

- ▶ Model $\Pr(y|x)$ parametrically
- ▶ Often defined as $\Pr(y|x) \propto \exp(w^\top \phi(x, y))$
- ▶ Inference remains $\arg \max_y w^\top \phi(x, y)$

Evaluating text classifiers

- ▶ Typically, many (thousands possible) classes
- ▶ A document may belong to several classes
- ▶ For any class, number of positive documents very small compared to number of negative documents
- ▶ Each document x has a set of associated classes Y_x
- ▶ For each class y and each document x , build a 2×2 matrix $M_{x,y}$ of booleans

$$M_{x,y}[0, 0] = [y \in Y_x \text{ and classifier outputs } y]$$

$$M_{x,y}[0, 1] = [y \in Y_x \text{ and classifier does not output } y]$$

$$M_{x,y}[1, 0] = [y \notin Y_x \text{ and classifier outputs } y]$$

$$M_{x,y}[1, 1] = [y \notin Y_x \text{ and classifier does not output } y]$$

Recall, precision, micro-, macro-average

- ▶ Micro-averaged contingency matrix is $M_\mu = \sum_{x,y} M_{x,y}$
- ▶ Microaveraged recall is $\frac{M_\mu[0,0]}{(M_\mu[0,0] + M_\mu[0,1])}$
- ▶ Microaveraged precision is $\frac{M_\mu[0,0]}{(M_\mu[0,0] + M_\mu[1,0])}$
- ▶ If there is class skew, macroaveraging may be preferable
- ▶ Compute per-class contingency matrix $M_y = \sum_x M_{x,y}$
- ▶ Scale so the four elements of M_y add up to 1
- ▶ Now average over classes
- ▶ Break-even, F_1 , as usual
- ▶ Other loss models: topics may be in tree/DAG

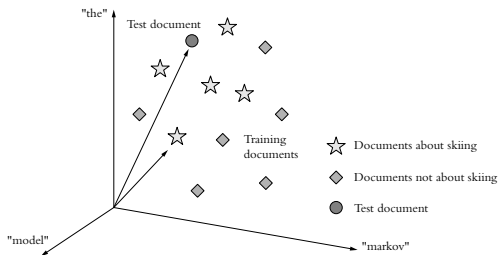
Outline

- ▶ Nearest-neighbor classification
- ▶ Bayesian classification
- ▶ Compression and minimum entropy criterion
- ▶ Maximum entropy and logistic regression
- ▶ Discriminative and max-margin classification

Nearest neighbor classification

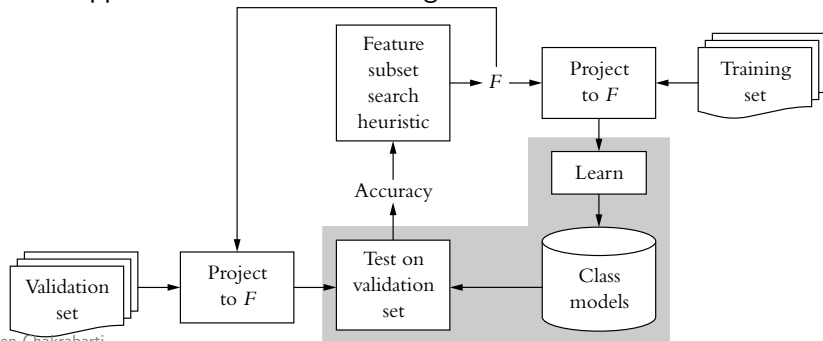
- ▶ Corpus indexed for TFIDF search
- ▶ Record class label with each (training) doc d
- ▶ Given a test doc q , find most similar docs d
- ▶ Take (weighted) majority vote of classes
- ⊕ Reuse existing infrastructure
- ⊖ IDF is not necessarily a good device for (soft) feature selection
 - ▶ w_1 appears at rate 0.01 uniformly across corpus
 - ▶ w_2 appears at rate 0.3 in one topic only
 - ▶ w_2 has lower IDF than w_1 but is a better feature

$$\arg \max_{\gamma} \sum_d \text{sim}(q, d) \mathbb{I}[d \text{ is labeled } \gamma]$$



Feature selection

- ▶ Unlike low-dimensional data mining and machine learning, the vector space model frequently has more dimensions than data points!
- ▶ Classifiers can **overfit** in presence of excessive number of noisy features
- ▶ A complicated subset search problem; typically solved by **accumulation** or **truncation**
- ▶ Wrapped around a basic learning black-box



Accumulation vs. truncation

Accumulation

- ▶ Accumulate: order features by decreasing correlation with class labels, pick a prefix
- ▶ Cheap and simple, but possible to include redundant features
- ▶ Classifier may not care (e.g. LR, SVM)

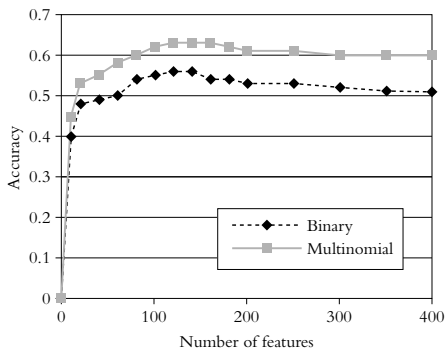
Truncation

- ▶ Start with all features $F = T$, drop X if X has a “good” Markov blanket $M \subseteq T \setminus X$
- ▶ I.e., X is “almost” conditionally independent of $(T \cup C) \setminus (M \cup X)$ given M
- ▶ Presence of M in F renders X unnecessary as a predictor
- ▶ May be no exact blankets, need heuristic ordering

Truncation example

- 1: Want $F \subset T$ the set of all terms
- 2: **while** truncated $\Pr(C|F)$ is reasonably close to original $\Pr(C|T)$ **do**
- 3: **for** each remaining feature X **do**
- 4: /* Identify a candidate Markov blanket M : */
- 5: For some tuned constant k , find the set M of k variables in $F \setminus X$ that are most strongly correlated with X
- 6: /* Estimate how good a blanket M is: */
- 7: Estimate $\sum_{x_M, x} \Pr(X_M = x_M, X = x)$
 $\text{KL}(\underbrace{C|X_M = x_M, X = x}_{\text{joint}} \parallel \underbrace{C|X_M = x_M}_{\text{marginal}})$
- 8: Eliminate the feature having the best surviving Markov blanket

Sample results



- ▶ Even highly biased learners (will explain) like naive Bayes can overfit
- ▶ Number of retained features typically set by cross-validation

Classification as compression

- ▶ Two topics, *Sports* and *Painting* with training document sets D_{Sports} and $D_{Painting}$
- ▶ Within each class concat documents and compress (gzip) and note down compressed lengths ℓ_{Sports} and $\ell_{Painting}$
- ▶ Given test document d , find compressed lengths
 - ▶ ℓ'_{Sports} of D_{Sports} concat d
 - ▶ $\ell'_{Painting}$ of $D_{Painting}$ concat d
- ▶ Compare $\ell'_{Sports} - \ell_{Sports}$ with $\ell'_{Painting} - \ell_{Painting}$
- ▶ Which class should we assign to d ?

The class-word contingency table

- ▶ n documents with (single) class labels (d_i, c_i)
- ▶ n_c is the number of documents in class c
- ▶ d_{iw} is the number of occurrences of word w in document i
- ▶ k_{cw} is the number of occurrences of word w over documents belonging to class c
- ▶ $K_{\cdot w} = \sum_c k_{cw}$, $K_{c \cdot} = \sum_w k_{cw}$
- ▶ $K = \sum_w K_{\cdot w} = \sum_c K_{c \cdot}$
- ▶ Marginal probability of class c is $p_c = K_{c \cdot} / K$
- ▶ Marginal probability of word w is $p_w = K_{\cdot w} / K$
- ▶ Conditional probability $p_{w|c} = k_{cw} / K_{c \cdot}$

Minimum entropy criterion

- ▶ For random variables A, B ,

$$H(A, B) = - \sum_{a,b} \Pr(a, b) \log \Pr(a, b)$$

$$H(A|B) = - \sum_{a,b} \Pr(a, b) \log \Pr(a|b)$$

$$H(A, B) = H(A) + H(B|A) = H(B) + H(A|B) \quad \text{▶ HW}$$

- ▶ In particular for words and classes

$$H(W, C) = - \sum_c p_c \log p_c - \sum_c p_c \sum_w p_{w|c} \log p_{w|c} \quad \text{▶ HW}$$

- ▶ Low entropy means less uncertainty, so we want small $H(W, C)$

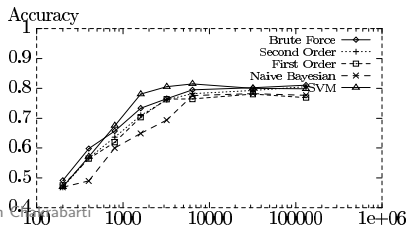
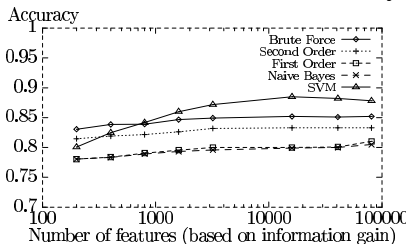
Classifying a test document

- ▶ As in the gzip example, pretend to add the test document to the corpus, each class in turn
- ▶ Suppose word counts in test document are x_w for word w , X total
- ▶ Suppose the tentative class is \hat{c}
- ▶ Update probability estimates:

$$p'_c = \begin{cases} \frac{K_{c\cdot} + X}{K + X}, & c = \hat{c} \\ \frac{K_{c\cdot}}{K + X}, & \text{otherwise} \end{cases}$$
$$p'_{w|c} = \begin{cases} \frac{k_{cw} + x_w}{K_{c\cdot} + X}, & c = \hat{c} \\ \frac{k_{cw}}{K_{c\cdot}}, & \text{otherwise} \end{cases}$$

Classifying a test document, continued

- ▶ Now compute new $H'_{\hat{c}}(W, C)$ using $p'_c, p'_{w|c}$
- ▶ Finally pick $\arg \min_{\hat{c}} H'_{\hat{c}}(W, C)$ as the predicted class
- ▶ Classification time is proportional to number of classes times number of words in vocabulary



Maxent notation setup

- ▶ Training examples (d_i, c_i) , $i = 1, \dots, n$
- ▶ $c_i \in C$, a flat set with 2 or more classes
- ▶ Goal is to directly model $\Pr(c|d)$
- ▶ Feature vector $\phi(d, c) \in \mathbb{R}^m$
- ▶ An element is $\phi_j(d, c) \in \mathbb{R}$, often 0/1 but not necessarily
- ▶ j can index over some feature space (to be discussed soon)
- ▶ We are interested in fitting a multinomial distribution over classes $\Pr(c|d_i)$ for each training document d_i
- ▶ This is just a table $p(c|d_i)$
- ▶ Later it will turn out that the model can be used with new documents as well

Balancing modeled and observed averages

- ▶ Empirical probability of document d_i is $1/n$
- ▶ Express $E(\phi_j)$ in two ways
 - ▶ $E(\phi_j) = \sum_{i=1}^n \widetilde{\Pr}(d_i, c_i) \phi_j(d_i, c_i) = \sum_{i=1}^n \frac{1}{n} \phi_j(d_i, c_i)$
 - ▶ $E(\phi_j) = \sum_{i=1}^n \widetilde{\Pr}(d) \sum_c \text{Pr}(c|d_i) \phi_j(d_i, c) = \sum_{i=1}^n \frac{1}{n} \sum_c \text{Pr}(c|d_i) \phi_j(d_i, c)$
- ▶ For a reasonable model $\text{Pr}(c|d_i)$, these should be equal:

$$\sum_{i=1}^n \frac{1}{n} \phi_j(d_i, c_i) = \sum_{i=1}^n \frac{1}{n} \sum_c \text{Pr}(c|d_i) \phi_j(d_i, c)$$

- ▶ Write $\text{Pr}(c|d_i)$ as a table of parameters $p(c|d_i)$ that we need to fit from data

$$\forall j : \quad \sum_{i=1}^n \phi_j(d_i, c_i) - \sum_{i=1}^n \sum_c p(c|d_i) \phi_j(d_i, c) = 0$$

The maximum entropy principle

- ▶ Many distributions $p(c|d_i)$ may satisfy the equalities
- ▶ Which one should we prefer?
- ▶ If we know nothing about d or class populations, the only reasonable $p(c|d)$ is the uniform distribution over $\{c\}$
- ▶ The uniform distribution has the largest possible entropy
- ▶ So we will maximize the entropy while satisfying the equalities

$$\max_{\{p(c|d_i) \geq 0\}} \quad - \sum_{i=1}^n \sum_c \underbrace{\widetilde{\text{Pr}}(d_i)}_{=1/n} p(c|d_i) \log p(c|d) \quad \text{subject to}$$

$$\forall j : \quad \sum_{i=1}^n \phi_j(d_i, c_i) - \sum_{i=1}^n \sum_c p(c|d_i) \phi_j(d_i, c) = 0 \quad \dots \lambda_j$$

$$\forall i : \quad \sum_c p(c|d_i) - 1 = 0 \quad \dots \mu_i$$

The dual solution

$$\begin{aligned} \text{Minimize } & L(\{p(c|d_i)\}; \{\lambda_j\}, \{\mu_i\}) \\ &= \sum_{i=1}^n p(c|d_i) \log p(c|d_i) \\ &\quad + \sum_j \lambda_j \left(\sum_{i=1}^n \phi_j(d_i, c_i) - \sum_{i=1}^n \sum_c p(c|d_i) \phi_j(d_i, c) \right) \\ &\quad + \sum_i \mu_i (\sum_c p(c|d_i) - 1) \end{aligned}$$

$$\frac{\partial L}{\partial p(c|d_i)} = 1 + \log p(c|d_i) - \sum_j \lambda_j \phi_j(d_i, c) + \mu_i = 0$$

$$\begin{aligned} \therefore p(c|d_i) &= \exp(-1 - \mu_i) \exp \left(\sum_j \lambda_j \phi_j(d_i, c) \right) \\ &= \frac{1}{Z(d_i)} \exp \left(\sum_j \lambda_j \phi_j(d_i, c) \right) = \frac{1}{Z(d_i)} \exp(\Lambda^\top \phi(d_i, c)), \end{aligned}$$

where $Z(d_i)$ is a scaling factor to make $\sum_c p(c|d_i) = 1$

Dual optimization

- ▶ Can show that maximizing entropy in primal is equivalent to maximizing $\sum_{i=1}^n \log \Pr(c_i|d_i)$
- ▶ With $\Pr(c|d_i)$ modeled as $p(c|d_i) = \frac{1}{Z(d_i)} \exp(\Lambda^\top \phi(d_i, c))$
- ▶ Therefore the optimization looks like

$$\max_{\Lambda} \sum_{i=1}^n \left(\Lambda^\top \phi(d_i, c_i) - \log Z(d_i) \right)$$

- ▶ Nontrivial because of the Z term
- ▶ Can be solved using Newton method to get trained Λ^*
- ▶ The prediction for a new test document d_{test} is just $\arg \max_c \Lambda^* \phi(d_{\text{test}}, c)$

Feature design

- ▶ Focus on the term $\Lambda \cdot \phi(d, c) = \sum_{j=1}^J \lambda_j \phi_j(d, c)$
- ▶ First we design a feature space compatible with naive Bayes
- ▶ Recall in naive Bayes, there is a parameter $\theta_{c', w}$ for each class c' and each word w
- ▶ We will make the number of features and model weights $J = CW$
- ▶ Let j range over c', w where c' is a class and w is a word
- ▶ A simple binary feature would be $\phi_{c', w}(d, c) = \mathbb{I}[c = c'] \mathbb{I}[w \in d]$
- ▶ With $d \in \{0, 1\}^W$, $\phi(d, c) \in \{0, 1\}^{C \times W}$
- ▶ Can think of $\phi(d, c)$ as a $C \times W$ 'matrix'
- ▶ c th row is just d , other rows are all zeroes
- ▶ Consider row class 'sports' and column word 'football': we expect $\phi_{\text{sports}, \text{football}}(d, \text{sports})$ to be 1 often
- ▶ Meanwhile $\phi_{\text{arts}, \text{football}}(d, \text{sports}) = 0$

Feature design (2)

- ▶ Also, we expect $\phi_{\text{arts,football}}(d, \text{arts})$ to be 0 often
- ▶ Note how ϕ is ‘called’ in the objective: with gold class as $\phi(d_i, c_i)$ and with all (other) classes as $\phi(d_i, c)$

$$\Lambda \cdot \phi(d_i, c_i) - \log \left[\sum_c \exp(\Lambda \cdot \phi(d_i, c)) \right]$$

- ▶ To maximize this, $\lambda_{\text{sports,football}} \uparrow\uparrow$ and $\lambda_{\text{arts,football}} \downarrow\downarrow$
- ▶ Just like we expect $\theta_{\text{sports,football}}$ to be larger than $\theta_{\text{arts,football}}$
- ▶ But we need not be limited to word-based (‘lexical’) feature spaces
- ▶ An example of a different kind of feature j is “mentions two cities and years and topic is history”
- ▶ Or, “mentions a dollar amount and is spam mail”
- ▶ These synthetic features can coexist side-by-side with lexical features
- ▶ Unlike naive Bayes, highly correlated features do not damage posterior label probability estimates (as badly)

Maxent and Logistic Regression (LR)

- ▶ Two-class case with $C = \{-1, +1\}$, e.g., *Sports* and *NotSports*
- ▶ Vocabulary of size W
- ▶ j ranges over $2W$ index positions, and $\Lambda \in \mathbb{R}^{2W}$
- ▶ Say the first W elements correspond to $\underline{c} = -1$ and the next W elements correspond to $\underline{c} = +1$, so that we can write $\Lambda = (\Lambda_+, \Lambda_-)$

- ▶ Given our canonical feature design $\phi_{\underline{c},w}(d, c)$

$$\Lambda^\top \phi(d, +1) = \Lambda_+^\top \psi(d) \quad \text{and}$$

$$\Lambda^\top \phi(d, -1) = \Lambda_-^\top \psi(d), \quad \text{where}$$

$$\psi(d) \in \mathbb{R}^W \quad \text{with} \quad \psi_w(d) = \mathbb{I}[w \in d].$$

Maxent and Logistic Regression (LR) (2)

- ▶ Now we can write

$$\begin{aligned}\Pr(+1|d) &= \frac{\exp(\Lambda^\top \phi(d, +1))}{\exp(\Lambda^\top \phi(d, +1)) + \exp(\Lambda^\top \phi(d, -1))} \\ &= \frac{\exp(\Lambda_+^\top \psi(d))}{\exp(\Lambda_+^\top \psi(d)) + \exp(\Lambda_-^\top \psi(d))} \\ &= \frac{1}{1 + e^{-(\Lambda_+ - \Lambda_-)^\top \psi(d)}} = \frac{1}{1 + e^{-\beta^\top \psi(d)}}, \quad \text{say}\end{aligned}$$

- ▶ Logistic regression: fit β to maximize $\sum_i \log \Pr(c_i|d_i)$
- ▶ Under infinite precision arithmetic, if there exists a β^* such that for all i , $c_i \beta^{*\top} \psi(d_i) > 0$, then the optimizer will bomb ▶ HW
- ▶ To control this, find instead

$$\arg \max_{\beta} -\frac{1}{2\sigma^2} \|\beta\|_2^2 + \sum_i \log \Pr(c_i|d_i)$$

- ▶ Equivalent to imposing a Gaussian prior on β with zero mean and variance σ^2

Linear discriminants

- ▶ Two classes ± 1
- ▶ Naive Bayes compares

$$\pi_{-1} \prod_w \theta_{-1,w}^{x_w} :: \pi_1 \prod_w \theta_{1,w}^{x_w}, \quad \text{i.e.,}$$

$$\log \pi_{-1} + \sum_w x_w \log \theta_{-1,w} :: \log \pi_1 + \sum_w x_w \log \theta_{1,w}, \quad \text{i.e.,}$$

$$0 :: \log \frac{\pi_1}{\pi_{-1}} + \sum_w x_w \log \frac{\theta_{1,w}}{\theta_{-1,w}}$$

- ▶ Final form $\beta^\top x \leq 0$
- ▶ Same can be easily verified for maxent/logistic

Outline, continued

- ▶ Discriminative and max-margin classification
- ▶ Parameter shrinkage in topic hierarchies
- ▶ Bayesian and SVM classifiers for topic hierarchies
- ▶ Modeling associative hyperlinks for max-margin classifiers
- ▶ Loopy belief propagation in hyperlink graphs

From maxent/LR to SVM

- ▶ Two-class case $c_i \in \pm 1$

- ▶ If $c_i = +1$, want $\beta \cdot \psi(d_i) > 0$

- ▶ If $c_i = -1$, want $\beta \cdot \psi(d_i) < 0$

- ▶ I.e., want to directly find a β such that

$$\forall i : \quad c_i \beta^\top \psi(d_i) \geq 0$$

without any probabilistic connotation

- ▶ Add a margin

$$\forall i : \quad c_i \beta^\top \psi(d_i) \geq 1$$

- ▶ Regularization remains similar: minimize $\|\beta\|_2^2$

- ▶ Overall optimization

$$\min_{w, \xi \geq 0} \frac{1}{2} w^\top w + \frac{B}{n} \sum_i \xi_i \quad \text{s.t.}$$

$$\forall i : \quad c_i \beta^\top \psi(d_i) + \xi_i \geq 1$$

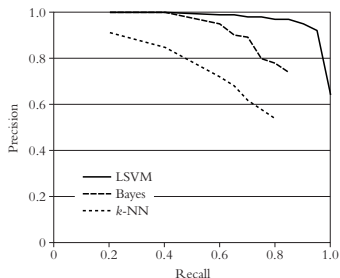
Comparison

- In the end, naive Bayes, maxent/LR, and SVM all give linear discriminants

$$c^* = \arg \max_c \beta^\top \psi(d) = \text{sign}(\beta^\top \psi(d))$$

for two classes $\{-1, +1\}$

- Why is maxent/LR (and SVM) so much better than naive Bayes?



Multiclass, single label per instance

- ▶ More than two topics, $|\mathcal{Y}| > 2$
- ▶ Train an SVM for each topic $y \in \mathcal{Y}$, with model weights $w_y \in \mathbb{R}^D$
- ▶ Document featurized into $x \in \mathbb{R}^D$
- ▶ Predict single class $\operatorname{argmax}_y w_y \cdot x$
- ▶ During training with instance (x_i, y_i) , want $w_{y_i} \cdot x_i \gg w_{y'} \cdot x_i$ for each $y' \neq y_i$
- ▶ I.e. good class score beats each bad class score
- ▶ With fixed margin 1, we can demand $w_{y_i} \cdot x_i \geq 1 + w_{y'} \cdot x_i$ for each $y' \neq y_i$
- ▶ In case not feasible, add slack variables $\xi_i \geq 0$ for each training instance: $w_{y_i} \cdot x_i + \xi_i \geq 1 + w_{y'} \cdot x_i$ for each $y' \neq y_i$
- ▶ And charge in the objective to be minimized as $\sum_i \xi_i$

Multiclass, single label per instance (2)

- ▶ Equivalent to the following hinge loss

$$\sum_i \max \left\{ 0, 1 + \max_{y' \neq y_i} w_{y'} \cdot x_i - w_{y_i} \cdot x_i \right\}$$

- ▶ Same margin of 1 for all y_i, y' pairs may be unsatisfactory
- ▶ $y_i = \text{photography}, y' = \text{painting}$ is less serious error than $y' = \text{rugby}$
- ▶ Seriousness of mistakes must be provided by trainer as $|\mathcal{Y}| \times |\mathcal{Y}|$ loss matrix $\Delta(y, y') \geq 0$
- ▶ Modify above loss to

$$\sum_i \max \left\{ 0, \max_{y' \neq y_i} w_{y'} \cdot x_i + \Delta(y_i, y') - w_{y_i} \cdot x_i \right\}$$

- ▶ This objective is convex, unlike the expected loss in case of logistic regression

$$\sum_i \sum_y \Delta(y_i, y) \Pr(y|x_i; \Lambda)$$

Multiple classes and labels

- ▶ More than two topics, $|\mathcal{Y}| > 2$
- ▶ Each document can have multiple labels
- ▶ Can still use one vs. rest, but it won't (sufficiently) exploit correlations between labels
- ▶ Structured prediction: $\mathbb{Y} = 2^{\mathcal{Y}}$
- ▶ Can use Hamming or Jaccard loss:

$$\Delta(y, y') = |y \setminus y'| + |y' \setminus y|$$

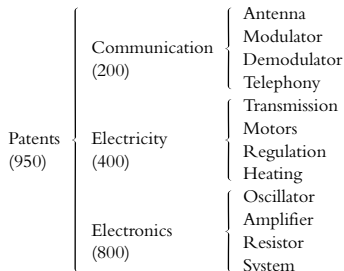
$$\Delta(y, y') = |y \cap y'| / |y \cup y'|$$

where $y, y' \in \mathbb{Y}$ are subsets of topics

- ▶ Design of $\phi(x, y)$
- ▶ If $x \in \mathbb{R}^d$, let $\phi(x, y) \in \mathbb{R}^{d|\mathcal{Y}|}$
- ▶ If $\gamma \in y$, copy x to the γ th block of ϕ
- ▶ Enumerating through \mathbb{Y} is not practical, how to train/test?

Topic hierarchy

- ▶ Very common in document classification
- ▶ For now, trees only, assume subsumption
- ▶ May hurt if we take a sequence of unreliable decisions
- ▶ May help in parameter estimation and scalability



Classifier	Flat	Best-First
Number of features	250	Shown in parentheses
Approximate total number of parameters	2651	2649
Accuracy	0.60	0.63
Time/document	15 ms	6 ms

Probabilistic classification into hierarchy

- 1: Initialize frontier min-heap F to $\{\langle \text{root}, 0 \rangle\}$
- 2: Initialize output leaf set L to the empty set
- 3: **while** $|L| < m$ and F is not empty **do**
- 4: Remove $\langle c_0, \ell_0 \rangle$ from F with smallest ℓ
- 5: **if** c_0 is a leaf node **then**
- 6: Insert c_0 into L
- 7: **else**
- 8: **for** each child c_i of c_0 **do**
- 9: Evaluate $\text{Pr}(c_i|c_0, d)$
- 10: Insert $\langle c_i, \ell_0 - \log \text{Pr}(c_i|c_0, d) \rangle$ into F
- 11: **Return** L

Parameter smoothing

- ▶ Uniform prior and Laplace smoothing
- ▶ Lidstone smoothing
- ▶ Beta prior

From smoothing to shrinkage

- ▶ Given a hierarchy, uniform prior is suboptimal, loses info
- ▶ Nodes near the top have plenty of training data, low variance
- ▶ Nodes near the leaves with sparse training data can benefit from estimates higher up

$$\tilde{\theta}_{c_i,t} = \lambda_i \theta_{c_i,t}^{\text{MLE}} + \dots + \lambda_1 \theta_{c_1,t}^{\text{MLE}} + \lambda_0 \theta_{c_0,t}^{\text{MLE}}$$

where $\sum \lambda_i = 1$

- ▶ λ s estimated by an EM-like procedure

Shrinkage pseudocode

- 1: hold out some portion H_n of the training set for class c_n
- 2: using the remaining data find $\theta_{c_i,t}^{\text{MLE}}$ for $i = 1, \dots, n$
- 3: initialize all λ_i to some positive value so that $\sum_i \lambda_i = 1$
- 4: **while** $\text{Pr}(H_n | \{\theta_{c_n,t} \forall t\})$ increases **do**
- 5: **for** $i = 0, 1, \dots, n$ **do**
- 6: Calculate $\beta_i = \sum_{t \in H_n} \frac{\lambda_i \theta_{c_i,t}^{\text{MLE}}}{\sum_j \lambda_j \theta_{c_j,t}^{\text{MLE}}}$, the degree to which the current estimate of class i predicts terms in the held-out set
- 7: readjust the mixing weights $\lambda_i \leftarrow \beta_i / \sum_j \beta_j$ for $i = 0, 1, \dots, n$
- 8: recompute $\tilde{\theta}_{c_n,t}$ for all t

Maxent/logistic on trees

- ▶ For notational simplicity assume all paths to leaves in tree have same length H
- ▶ Label y is now a sequence of H nodes in tree $y^1, \dots, y^h, \dots, y^H$ (y^1 is always root)
- ▶ \mathcal{L} is the set of leaf nodes, in 1:1 correspondence to possible y paths
- ▶ \mathcal{Y} is the set of all nodes in label tree
- ▶ Think similar to an HMM: label at each tree level is a **state**
- ▶ **Edge potentials** $\psi_h(\gamma, \gamma')$
- ▶ $\psi_h(\gamma, \gamma') = 0$ unless γ is a state at level h , γ' at level $h + 1$, and γ a parent of γ'
- ▶ Same doc x **emitted** at each level
- ▶ **Node potentials** $\varphi_h(x, \gamma)$
- ▶ $\varphi_h(x, \gamma) = 0$ unless γ is a label at level h

Maxent/logistic on trees (2)

- ▶ Otherwise, expresses compatibility between x and γ
- ▶ Overall probability modeled as

$$\Pr(y|x) = \frac{1}{Z_x} \prod_{h=1}^{H-1} \psi_h(y^h, y^{h+1}) \prod_{h=1}^H \varphi_h(x, y^h)$$

$$\text{where } Z_x = \sum_y \prod_{h=1}^{H-1} \psi_h(y^h, y^{h+1}) \prod_{h=1}^H \varphi_h(x, y^h)$$

- ▶ Learnable parameters α, β inside ψ, φ
- ▶ E.g., $\alpha \in \mathbb{R}^{|\mathcal{Y}| \times |\mathcal{Y}|}$, with $\psi_h(y^h, y^{h+1}) = \exp(\alpha(y^h, y^{h+1}))$
$$\prod_{h=1}^{H-1} \psi_h(y^h, y^{h+1}) = \prod_{h=1}^{H-1} \exp(\alpha(y^h, y^{h+1})) = \exp\left(\sum_{h=1}^H \alpha(y^h, y^{h+1})\right)$$
- ▶ Abusing notation, this can be written in a log-linear form $\exp(\alpha^\top g(x, y))$ (x is not used)

Maxent/logistic on trees (3)

- ▶ Likewise, $\prod_h \varphi_h(x, y_h)$ can be expressed in a log-linear form $\exp(\beta^\top f(x, y))$

- ▶ Can combine into single $w^\top \phi(x, y)$, i.e.,

$$\Pr(y|x) = \frac{1}{Z_x} \exp(w^\top \phi(x, y))$$

- ▶ w should embed learnable parameters that capture associations between
 - ▶ parent and child classes
 - ▶ class and word
- ▶ Rest is same as in standard logistic regression:
 $\arg \max_w \sum_i \log \Pr(y_i | x_i)$

Label trees in SVM

- ▶ Redesign loss Δ and feature map ϕ
- ▶ What is y ? A path from root to leaf, say
- ▶ $\Delta(y, y')$ compares two paths
- ▶ If paths diverge early, loss is larger
- ▶ $\phi(x, y)$ embeds shrinkage-like ideas
- ▶ **Key idea: Also map labels to vector space** $\psi(y) \in \mathbb{R}^k$
- ▶ Familiar special case with $k = |\mathcal{Y}|$ and $\psi_\gamma(y) = \mathbb{I}[\gamma = y]$

$$\phi(x, y) = \begin{bmatrix} \psi_1(y)x \\ \vdots \\ \psi_k(y)x \end{bmatrix} \in \mathbb{R}^{dk}$$

- ▶ Aka $\phi(x, y) = \psi(y) \otimes x$ (tensor product)
- ▶ $w \in \mathbb{R}^{dk}$ as well, naturally

Label trees in SVM (2)

- ▶ Choice of ψ for label trees? Let $k = |\mathcal{Y}|$ and $0 < \lambda \leq 1$

$$\psi_{\gamma}(y) = \begin{cases} \lambda, & \gamma \text{ is an ancestor of } y \\ 0, & \text{otherwise} \end{cases}$$

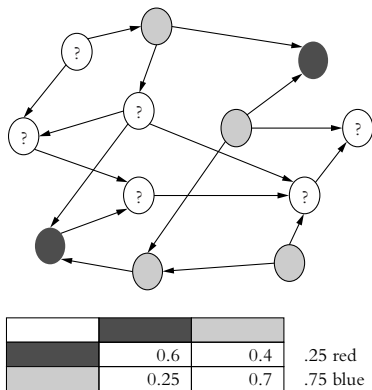
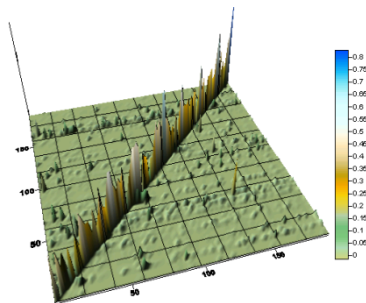
- ▶ Or may be λ raised to a power that is the path length between γ and y ?
- ▶ Let w_{γ} be the block of w corresponding to label γ
- ▶ Easy to see that

$$w^{\top} \phi(x, y) = \sum_{\gamma} \psi_{\gamma}(y) w_{\gamma}^{\top} x$$

- ▶ Intuitive: borrow feature vector x into positions y guided by ancestor-descendant relation
- ▶ Dual training etc. goes through fine

Labeling Web documents

- ▶ Many Web documents are short compared to IR corpora
- ▶ Depend on **hyperlinks**, images, flash, Javascript, to lead the user to content
- ▶ Topics are often **associative** across hyperlinks (birds of a feather)
- ▶ Key parameter: $\Pr(\text{neighbor red} | \text{I am blue})$



Associative model

- ▶ First, an undirected model
- ▶ $f : V \rightarrow Y$ is a labeling of nodes
- ▶ $L(u, f(u))$ is the cost of assigning label $f(u)$ to node u
- ▶ Think $L(u, y) = -\log \Pr(Y_u = y)$
- ▶ **Affinity matrix** $A(\gamma, \gamma')$, $A \in \mathbb{R}_+^{|\mathcal{Y}| \times |\mathcal{Y}|}$
- ▶ If $\{u, v\} \in E$, you pay $A(f(u), f(v))$
- ▶ Overall optimization

$$\arg \min_f \sum_{u \in V} L(u, f(u)) + \sum_{\{u, v\} \in E} A(f(u), f(v))$$

- ▶ Also, marginal probability of red and blue
- ▶ A coloring problem with soft constraints
- ▶ NP-hard
- ▶ All topics link to Yahoo!

Smoothing on edges

- ▶ M labels, N nodes
- ▶ Feature vector of node n is $\mathbf{x}_n \in \mathbb{R}^D$, collectively $\mathbf{X} \in \mathbb{R}^{N \times D}$
- ▶ In the absence of edges, might use multiclass Logistic regression $\mathbf{y}_n = \text{softmax}(\mathbf{x}_n \mathbf{W})$ with $\mathbf{W} \in \mathbb{R}^{D \times M}$
- ▶ Collectively, $\mathbf{Y} = \text{softmax}(\mathbf{X} \mathbf{W})$ where softmax normalization applies separately over each row
- ▶ Gold labels of all nodes in $N \times M$ matrix \mathbf{Y}^* with 1-hot rows
- ▶ Want small prediction error $\sum_n L_{\text{node}}(\mathbf{y}_n, \mathbf{y}_n^*)$
- ▶ Node adjacency matrix $\mathbf{A} \in \mathbb{R}_+^{N \times N}$ with non-negative edge weights
- ▶ Also want roughness $\sum_{i,j} A(i,j) L_{\text{edge}}(\mathbf{y}_i, \mathbf{y}_j)$ to be small

▶ HW Works out nicely for some choices of L_{node} and L_{edge} , e.g.

$$L_{\text{node}}(\mathbf{y}, \mathbf{y}^*) = \sum_{m \in [M]} (y[m] - y^*[m])^2 \text{ and}$$

$$L_{\text{edge}}(\mathbf{y}_i, \mathbf{y}_j) = \sum_m (y_i[m] - y_j[m])^2$$

A belief propagation approach

- ▶ $V^K \subset V$ are nodes with known labels $f(v)$
- ▶ u_T is the (known) text of node u ; V^T is all node text
- ▶ Bayesian goal would be to
$$\max_f \Pr(f(V)) \Pr(E, \{u_T, u \in V\} | f(V))$$
- ▶ $N(v)$ are immediate neighbors of v
- ▶ $N^U(v) \dots$ with unknown labels

$$\begin{aligned} & \Pr(f(v) | E, V^T, f(V^K)) \\ &= \sum_{f(N^U(v))} \Pr(f(v), f(N^U(v)) | E, V^T, f(V^K)) \\ &= \sum_{f(N^U(v))} \Pr(f(N^U(v)) | E, V^T, f(V^K)) \\ & \quad \Pr(f(v) | f(N^U(v)), E, V^T, f(V^K)) \end{aligned}$$

A belief propagation approach (2)

- ▶ Naive-Bayes approximation

$$\Pr(f(N^U(v)) | E, V^T, f(V^K)) \approx \prod_{w \in N^U(v)} \Pr(f(w) | E, V^T, f(V^K))$$

- ▶ Iterative scheme

$$\frac{\Pr_{(r+1)}(f(v) | E, V^T, f(V^K))}{\sum_{f(N^U(v)) \in \Omega_v} \left[\prod_{w \in N^U(v)} \frac{\Pr_{(r)}(f(w) | E, V^T, f(V^K))}{\Pr(f(v) | f(N^U(v)), E, V^T, f(V^K))} \right]}$$

A belief propagation approach (3)

- ▶ Next, simplify $\Pr\left(f(v) \mid f(N^U(v)), E, V^T, f(V^K)\right)$ via a Markov assumption

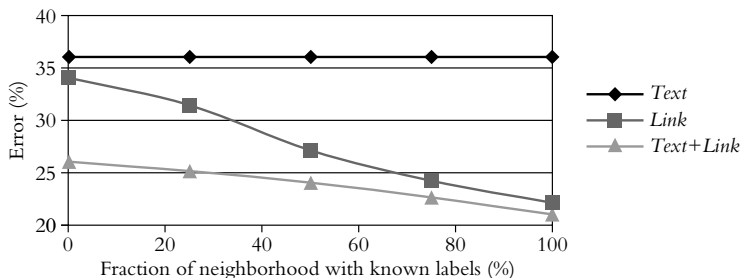
$$\begin{aligned} & \Pr\left(f(v) \mid f(N^U(v)), E, V^T, f(V^K)\right) \\ & \approx \Pr\left(f(v) \mid f(N^U(v)), E, V^T, f(N^K(v))\right) \\ & = \Pr\left(f(v) \mid f(N(v)), V^T\right), \end{aligned}$$

where $N^K(v) = N(v) \cap V^K$

- ▶ And finally decouple from other nodes' text

$$\Pr(f(v) \mid f(N(v)), V^T) \approx \Pr(f(v) \mid f(N(v)), v_T)$$

A belief propagation approach (4)



- ▶ Reduces error, scales gracefully with V^K
- ▶ Even when $V^K = \emptyset$, can improve accuracy (how?)
- ▶ Can also use direct conditional models (Lu and Getoor)
- ▶ Or a general graphical model, but costly