Maximum Entropy

> conditional probability $\implies$ maximize conditional likelihood

$p(y \mid x_1 \ldots x_n) \neq p(y_1 x_1 \ldots x_n)$ $\quad$ ↳ not just counts but optimization process

$$p_\lambda(y \mid \mathbf{x}) = \frac{e^{\sum_i \lambda_i f_i(y,x)}}{\sum_{y'=y} e^{\sum_i \lambda_i f_i(y',x)}}$$

$$= \frac{1}{Z(x)} \cdot e^{\sum_i \lambda_i f_i(y,x)}$$

same constant for each y (only depends on x)

f_i = feature – 0 or 1 – property of document x with class y
e.g. occurrence of the word "money" + class = spam → 1

λ_i = weight for feature f_i, can be high/positive (= count toward class y), low (= not expressive wrt class y) or negative (= count against y)

y = class
x = document vector (e.g. bag of words)

$d_1:$ "Where are they Pierre"
$d_2:$ "Where will you go"

LEARNING λ – WEIGHTS

> maximize the conditional log likelihood of the data given the model

$$\max_\lambda \frac{1}{\lambda} \log (p_\lambda(Y \mid X)) = \sum_{(y,x) \in (Y,X)} \log (p_\lambda(y \mid x))$$

$$= \sum_{(y,x) \in (Y,X)} \log \left(\frac{e^{\sum_i \lambda_i f_i(y,x)}}{\sum_{y'} e^{\sum_i \lambda_i f_i(y',x)}} \right)$$
↳ log · exp cancel out

$$= \sum_{(y,x) \in (Y,X)} \log \left(\frac{e^{\sum_i \lambda_i f_i(y,x)}}{\sum_{y'} e^{\sum_i \lambda_i f_i(y',x)}} \right)$$
logarithm function

$$\Leftrightarrow \sum_{(y,x) \in (Y,X)} \left(\log \exp \sum_i \lambda_i f_i(y,x) - \log \sum_{y'} \exp \sum_i \lambda_i f_i(y',x) \right)$$
reformulated against log rules

$$\Leftrightarrow \underbrace{\sum_{(y,x) \in (Y,X)} \log \exp \sum_i \lambda_i f_i(y,x)}_{A_\lambda} - \underbrace{\sum_{(y,x) \in (Y,X)} \log \sum_{y'} \exp \sum_i \lambda_i f_i(y',x)}_{B_\lambda}$$

find derivative of this and optimize = maximize

$\implies$ compute derivative of A and B separately

$$\frac{\partial A_\lambda}{\partial \lambda_i} = \sum_{(y,x) \in (Y,X)} f_i(y,x)$$
$\implies$ empirical feature count
model predicts perfect model = 1 the same words are empirically more

$$\frac{\partial B_\lambda}{\partial \lambda_i} = \sum_{(y,x) \in (Y,X)} \sum_{y'} p_\lambda(y' \mid x) f_i(y',x)$$
$\implies$ predicted feature count
$p_\lambda = 1$ if $y' = y$

training instance
number of correctly switched on training instances minus weighted sum of all switched on instances

FEATURE SELECTION

Mutual Information

$$I(X,y) = \sum_{y \in Y} \sum_{x \in X} p(x,y) \log \left(\frac{p(x,y)}{p(x) p(y)} \right)$$
↓ Labels $\quad$ feature values (e.g. words) $\quad \implies$ relative frequencies

$\implies$ PMI pointwise MI calculated importance of words, take X most important features

(right side – blue/purple boxes)

class Feature:
- property LABEL
- property property
- method setLabel
- method setProperty

} are used boolean to switch weights on or off if feature label and doc [feature property]

class MaxEnt:
- property features $\quad$ → list/dict with all features
- property weights $\quad$ → some amount, maybe amount of features learned in learnFeatures and weights initialized as a list with same length?
- method learnFeatures
- method train

after learning X number of features, initialize list of X number of weights between +2 and -2 ?

1. compute probabilities for all instances
2. for every weight:
   count all training instances switched on by the property $\quad$ if doc [property i]: $\partial \Theta_i$ += 1
   ∂A → multiply by respective probability
   ∂B → of these, count those also switched on by label $\quad$ if label (doc) == label: ∂A_i += 1
   $\implies A_i - B_i$ → update λ_i
3. update λ's
$\implies$ repeat until convergence

take some data (not training data?), calculate $p(w,y)$, $p(w)$, $p(y)$ for each word and label, maybe even $p(w,y)$, choose X number "most informative" features / highest PMI scores
↳ $\log \left(\frac{p(w,y)}{p(w) p(y)} \right)$

1. for each document: $\quad$ → index in doc vector
   count all words in it $\quad$ → vector steps over all docs
   a. alone $\quad$ → how to sum up lists elementwise $\quad$! only works for 1-0-0-1-... doc vectors (only sum occurrence once per doc)
   b. with the doc's label
   (c) not in it alone + with the label $\quad$ → dict of all labels, each label has its own vector sum list

2. compute $p(w,y)$ – for each combination
   as a simple $f(w,y)/f(w)$?
   $f(w)$/no of docs and $f(y)$/no of docs

3. order descending by log PMI
   $\implies$ return X number of features
   either • features above PMI - score of X
   • X number of highest scoring features overall
   • X number " " " per label