NATURAL LANGUAGE PROCESSING

Text Classification

TOPICS

- **Text Classification**
  - Naïve Bayes
  - Logistic Regression
  - Practical Issues

GENERATIVE VS. DISCRIMINATIVE

- Generative classifier
  - Build a model of how a class could generate some input data. Given an observation, they return the class most likely to have generated the observation.
- Discriminative classifiers:
  - Learn what features from the input are most useful to discriminate between the different possible classes.
- Discriminative systems are often more accurate and hence more commonly used.

GENERATIVE VS. DISCRIMINATIVE

- In the text categorization scenario, to predict whether a document belongs to the class c, ***P(c|d)***
  - Naive Bayes makes use of the likelihood term, which expresses how to generate the features of a ***document*** if we knew it was of ***class c***.
  - Logistic Regression attempts to directly compute ***P(c|d)*** by assigning high weight to document features that directly improve its ability to discriminate between possible classes,

TOPICS

- Text Classification
  - Naïve Bayes
  - **Logistic Regression**
  - Practical Issues

Yanjun Li

5

LOGISTIC REGRESSION

- A probabilistic classifier
  - The baseline supervised NLP algorithm
- Has a very close relationship with neural networks.
- Binary classifier or multinomial classifier

Yanjun Li

6

COMPONENTS OF LR

- A feature representation of the input.
  - A vector of features [x1,x2,...,xn].
- A classification function computes the estimated class, via ***P(y|x).***
  - the sigmoid and softmax tools for classification.
- An objective function for learning.
  - The cross-entropy loss function
- An algorithm for optimizing the objective function.
  - Stochastic gradient descent algorithm.

Yanjun Li 7

TWO PHASES OF LR

- Training
  - Train the system using stochastic gradient descent and the cross-entropy loss.
- Test
  - Compute p(y|x) and return the higher probability label y = 1 or y = 0.

Yanjun Li 8

FEATURE REPRESENTATION

- For specific NLP task, features are designed based on linguistic intuitions and the linguistic literature on the domain.
- Error analysis on the training set/dev. set of the early version of the system will provide insights into features.
- Hand-designed complex features (combination of primitive features) are helpful.
- Automatic representation learning

Yanjun Li

9

SENTIMENT CLASSIFICATION EXAMPLE

Var	Definition	Value in Fig. 5.2
x_1	count(positive lexicon) $\in$ doc)	3
x_2	count(negative lexicon) $\in$ doc)	2
x_3	$\begin{cases} 1 & \text{if "no"} \in \text{doc} \\ 0 & \text{otherwise} \end{cases}$	1
x_4	count(1st and 2nd pronouns $\in$ doc)	3
x_5	$\begin{cases} 1 & \text{if "!"} \in \text{doc} \\ 0 & \text{otherwise} \end{cases}$	0
x_6	log(word count of doc)	$\ln(64) = 4.15$

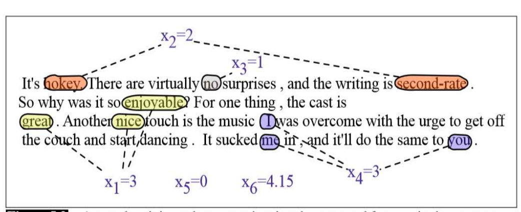


Figure 5.2 A sample mini test document showing the extracted features in the vector x.

Yanjun Li

10

PERIOD DISAMBIGUATION EXAMPLE

$$x_1 = \begin{cases} 1 & \text{if “}Case(w_i) = \text{Lower”} \\ 0 & \text{otherwise} \end{cases}$$

❌ U.S.A.

$$x_2 = \begin{cases} 1 & \text{if “}w_i \in \text{AcronymDict”} \\ 0 & \text{otherwise} \end{cases}$$

❌ Prof.

$$x_3 = \begin{cases} 1 & \text{if “}w_i = \text{St. \& } Case(w_{i-1}) = \text{Cap”} \\ 0 & \text{otherwise} \end{cases}$$

❌ St. (Street)

SIGMOID CLASSIFIER

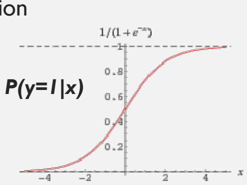
- Goal: “positive sentiment, P(y = 1|x) ” versus “negative sentiment, P(y = 0|x)”
- Learn a vector of weights ***w*** and bias term ***b*** (intercept)

$$\boldsymbol{z = w \cdot x + b}$$

- To create a probability, pass ***z*** to ***sigmoid*** function

$$\boldsymbol{y = \sigma(z) = 1/(1+e^{-z})}$$

class label is ***1 if P(y = 1|x) > 0.5 & P(y=0|x) = 1 – P(y=1|x)***



CROSS-ENTROPY LOSS FUNCTION

- Mean squared error loss function is hard to optimize (non-convex)
- Cross entropy loss function – the negative log likelihood loss

  $L_{CE}(\hat{y},y) = -\log P(y|x) = -[y \log \hat{y} + (1-y)\log(1-\hat{y})]$
  - The range is from zero (log of 1, no loss) to infinity (log of zero, infinite loss)
- The loss of the whole training sets is the sum of each training sample.
- The cost function is defined as the average loss for each sample.

Yanjun Li

13

GRADIENT FOR LOGISTIC REGRESSION

- The loss function is convex, so gradient descent is guaranteed to find the minimum – Walk downhill in the direction of the steepest slop.
- The slope is expressed as the partial derivative of the loss function is $(P(y|x) - y) \cdot X$ (**See SLP section 5.8 for details**)

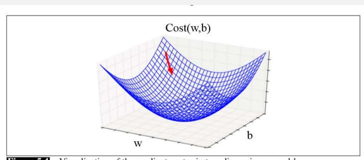


Figure 5.4 Visualization of the gradient vector in two dimensions *w* and *b*.

Yanjun Li

14

STOCHASTIC GRADIENT DESCENT ALGORITHM

- Three algorithms:
  - Batch Gradient Descent (average loss of all)
  - Mini-batch Gradient Descent (average loss of some)
  - ***Stochastic Gradient Descent*** (update W after minimize the loss of each sample)
- Learning rate should be adjusted, not too big, not too small.
  - It is most common to begin the learning rate at a higher value, and then slowly decrease it,

REGULARIZATION

- To avoid overfitting, a regularization term is added to the objective function.
- Two common terms R(w)
  - ***L1*** regularization – a linear function of the weight values, Manhattan distance. Complex, prefer vectors of fewer features, sparse solutions with large weights
  - ***L2*** regularization – a quadratic function of the weight values, Euclidean distance. Easier to optimize, prefer vectors with many small weights.

$$\hat{w} = \underset{w}{\operatorname{argmax}} \sum_{1=1}^{m} \log P(y^{(i)}|x^{(i)}) - \alpha R(w)$$

MULTINOMIAL LOGISTIC REGRESSION

- Softmax function is adopted to compute ***P(y=c|x)*** when ***c*** has more than two values.
  - Takes a vector of k arbitrary values and maps them to a probability distribution, each value in the range (0,1], and sum to 1.

$$\text{softmax}(z) = \left[\frac{e^{z_1}}{\sum_{i=1}^{k} e^{z_i}}, \frac{e^{z_2}}{\sum_{i=1}^{k} e^{z_i}}, \ldots, \frac{e^{z_k}}{\sum_{i=1}^{k} e^{z_i}}\right]$$

$$p(y=c|x) = \frac{e^{w_c \cdot x + b_c}}{\sum_{j=1}^{k} e^{w_j \cdot x + b_j}}$$

FEATURES IN MULTINOMIAL LR

- The input features are features regarding observations and the candidate output classes.
- A positive weight of a feature in multiclass is the evidence for or against an individual class.
- Example: 3 classes (+, -, neutral), the weight of feature !

Var	Definition	Wt
$f_1(0,x)$	$\begin{cases} 1 & \text{if "!"} \in \text{doc} \\ 0 & \text{otherwise} \end{cases}$	-4.5
$f_1(+,x)$	$\begin{cases} 1 & \text{if "!"} \in \text{doc} \\ 0 & \text{otherwise} \end{cases}$	2.6
$f_1(0,x)$	$\begin{cases} 1 & \text{if "!"} \in \text{doc} \\ 0 & \text{otherwise} \end{cases}$	1.3

APPLYING LOGISTIC REGRESSION TO BUILD BIGRAM LANGUAGE MODEL

- Word is represented as one-hot encoding vector (binary format of BOW feature vector).
- For bigram model,

  y = current_word (output), x=previous_word (input):

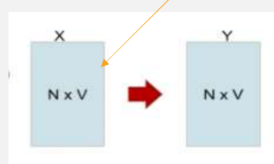
  P(y|x) = softmax(x·W)
- We find W (V x V) by doing gradient descent on the cost.

$$J = -\frac{1}{N}\sum_{n=1}^{N}\sum_{k=1}^{V} y_{n,k} log(p(y_{n,k}|x_n))$$

$$\textit{while } J \textit{ not converged}:$$
$$W \leftarrow W - \eta \nabla J, \textit{ where } \nabla J = X^T(p(Y|X) - Y)$$

- Implementation: LoR_bigram.ipynb

N is # of bigram, V is the size of vocabulary



NAÏVE BAYES VS. LOGISTIC REGRESSION

- Correlated Features
  - Due to strong conditional independence assumption, Naïve Bayes will overestimate the weight of them.
  - Logistic regression assigns part of weight to each.
- Small dataset or short documents
  - Naïve Bayes performs better.
- Implementation complexity
  - Naïve Bayes is easy to implement and fast to train.

TOPICS

- Text Classification
  - Naïve Bayes
  - Logistic Regression
  - **Practical Issues**

Sec. 15.3.1

NO TRAINING DATA? MANUALLY WRITTEN RULES

If (wheat or grain) and not (whole or bread) then

Categorize as grain

- Need careful crafting
  - Human tuning on development data
  - Time-consuming

Sec. 15.3.1

VERY LITTLE DATA?

- Use Naïve Bayes
- Get more labeled data
  - Find clever ways to get humans to label data for you
- Try semi-supervised training methods:
  - Bootstrapping, EM over unlabeled documents, …

Yanjun Li 23

Sec. 15.3.1

A REASONABLE AMOUNT OF DATA?

- Perfect for all the clever classifiers
  - SVM
  - Regularized Logistic Regression
- You can even use user-interpretable decision trees
  - Users like to hack
  - Management likes quick fixes

Yanjun Li 24

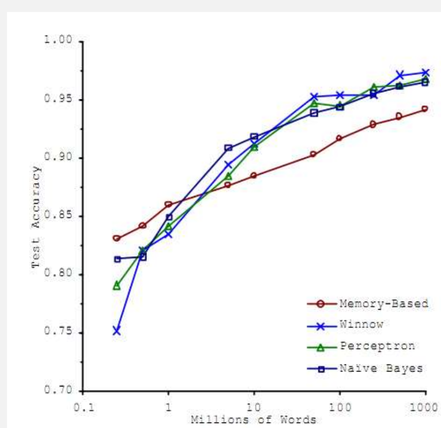
Sec. 15.3.1

A HUGE AMOUNT OF DATA?

- Can achieve high accuracy!
- At a cost:
  - SVMs (train time) or kNN (test time) can be too slow
  - Regularized logistic regression can be somewhat better

ACCURACY AS A FUNCTION OF DATA SIZE

- With enough data
  - Classifier may not matter



Brill and Banko on spelling correction

REAL-WORLD SYSTEMS GENERALLY COMBINE:

- Automatic classification
- Manual review of uncertain/difficult/"new" cases

UNDERFLOW PREVENTION: LOG SPACE

- Multiplying lots of probabilities can result in floating-point underflow.
- Since log(*xy*) = log(*x*) + log(*y*)
  - Better to sum logs of probabilities instead of multiplying probabilities.
- Class with highest un-normalized log probability score is still most probable.

$$c_{NB} = \underset{c_j \in C}{\operatorname{argmax}} \log P(c_j) + \sum_{i \in positions} \log P(x_i \mid c_j)$$

- Model is now just max of sum of weights

HOW TO TWEAK PERFORMANCE

- Domain-specific features and weights: *very* important in real performance
- Sometimes need to collapse terms:
  - Part numbers, chemical formulas, …
  - But stemming generally doesn't help ??
  - Upweighting: Counting a word as if it occurred twice:
    - title words (Cohen & Singer 1996)
    - first sentence of each paragraph (Murata, 1999)
    - In sentences that contain title words (Ko *et al,* 2002)