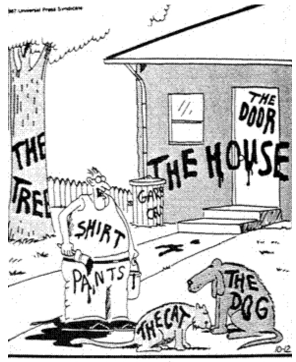
CS388: Natural Language Processing

Lecture 3: Multiclass Classification

Greg Durrett

TEXAS
The University of Texas at Austin



"Now! ... *That* should clear up a few things around here!"

Administrivia

- P1 due Tuesday, January 30 (one week)
- Anisha and Greg's OHs as normal this week (see course website)

Recall: Binary Classification

Logistic regression: $P(y=1|x) = \dfrac{\exp\left(\sum_{i=1}^{n} w_i x_i\right)}{\left(1+\exp\left(\sum_{i=1}^{n} w_i x_i\right)\right)}$ these sums are sparse!

Decision rule: $P(y=1|x) \geq 0.5 \Leftrightarrow w^\top x \geq 0$

Gradient: differentiate the log likelihood: $x(y - P(y=1|x))$

- This is the gradient of a single example. Can then apply stochastic gradient (or related optimization methods like Adagrad, etc.)
- ML pipeline: input -> feature representation, train model on labeled data (with stochastic gradient methods), then test on new data

This Lecture

- Evaluation in NLP (part 1)
- Multiclass fundamentals
- Feature extraction
- Multiclass logistic regression
- Start NNs (if time)

Evaluation in NLP

Evaluation in NLP

- For sentiment analysis: our evaluation was *accuracy*
- For more imbalanced classification tasks: accuracy doesn't make sense

Suppose we are classifying tokens as people's names or not:

The meeting was held between Barack Obama and Angela Merkel

The two heads of state discussed matters of the economy and the…

90+% of tokens will not be people's names depending on the text genre

Precision vs. Recall

- **Precision**: number of true positive predictions divided by number of positive predictions
- **Recall**: number of true positive predictions divided by total true positives

Predictions in blue, **ground truth in gold**

The meeting was held between Barack Obama and Angela Merkel

Precision = 2/3 = 0.66

Recall = 2/4 = 0.5

F1 or F-measure:
harmonic mean of these two = 0.57

Building Better Systems

System A: precision = 0.5, recall = 0.6, F1 = 0.55

System B: precision = 0.8, recall = 0.4, F1 = 0.53

- Which is better?

System A: precision = 0.5, recall = 0.6, F1 = 0.55

System B: precision = 0.51, recall = 0.61, F1 = 0.56

- Which is better?

Significance Tests

- Paired bootstrap: Suppose you have systems A and B and test set T. Hypothesis: perf(A, T) > perf(B, T)

```
stat = 0
for i in 0 to K        # number of trials
   T' ~ sample from T with replacement to create test set of the same size
   if perf(A, T') < perf(B, T')    # system performance flipped on T'
       stat += 1
return pvalue = stat/K
```

- Think about the size of your test set. If 100 examples, a 1% difference is 1 example. Is that really meaningful? This can check that!

Macro F1

- Suppose you have a multiclass classification problem with 10 classes
- Which system is better?

  Accuracy = 0.7, always predicts most frequent class
  Accuracy = 0.68, makes some correct predictions from every class

- *Macro-averaged F1 (Macro F1)*: compute F1 for each class (prec/rec of that class's labels), average these F1s

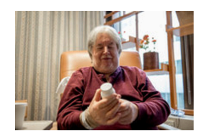
Multiclass Fundamentals

Text Classification

A Cancer Conundrum: Too Many Drug Trials, Too Few Patients

Breakthroughs in immunotherapy and a rush to develop profitable new treatments have brought a crush of clinical trials scrambling for patients.

By GINA KOLATA

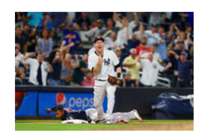


→ Health

Yankees and Mets Are on Opposite Tracks This Subway Series

As they meet for a four-game series, the Yankees are playing for a postseason spot, and the most the Mets can hope for is to play spoiler.

By FILIP BONDY



→ Sports

~20 classes

Image Classification

→ Dog

→ Car

- Thousands of classes (ImageNet)

Entailment

- Three-class task over sentence pairs
- Not clear how to do this with simple bag-of-words features

A soccer game with multiple males playing.
ENTAILS
Some men are playing a sport.

A black race car starts up in front of a crowd of people.
CONTRADICTS
A man is driving down a lonely road

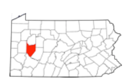
A smiling costumed woman is holding an umbrella.
NEUTRAL
A happy woman in a fairy costume holds an umbrella.

Bowman et al. (2015)

Entity Linking

Although he originally won the event, the United States Anti-Doping Agency announced in August 2012 that they had disqualified **Armstrong** from his seven consecutive Tour de France wins from 1999–2005.

Lance Edward Armstrong is an American former professional road cyclist



Armstrong County is a county in Pennsylvania…

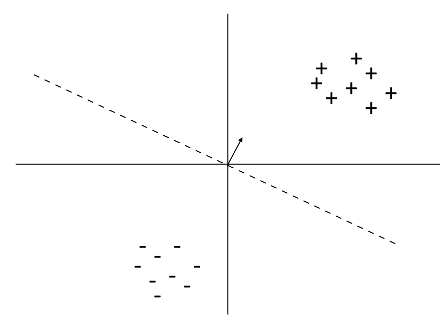
?

?

- 4,500,000 classes (all articles in Wikipedia)

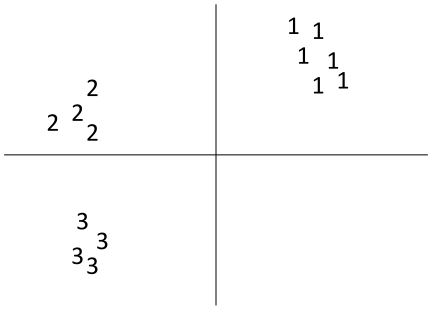
Binary Classification

- Binary classification: one weight vector defines positive and negative classes



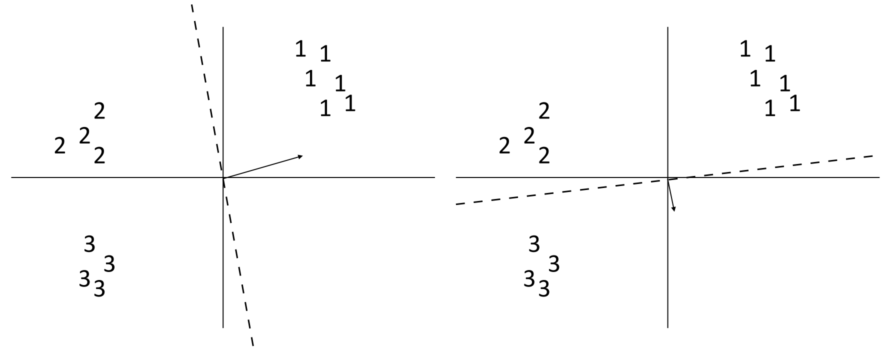
Multiclass Classification

- Can we just use binary classifiers here?



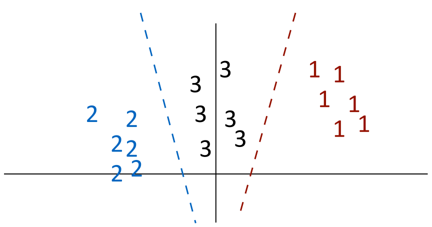
Multiclass Classification

- One-vs-all: train *k* classifiers, one to distinguish each class from all the rest
- How do we reconcile multiple positive predictions? Highest score?



Multiclass Classification

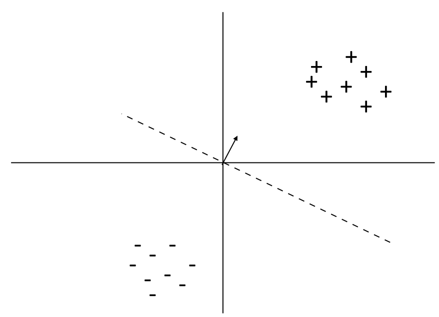
- Not all classes may even be separable using this approach



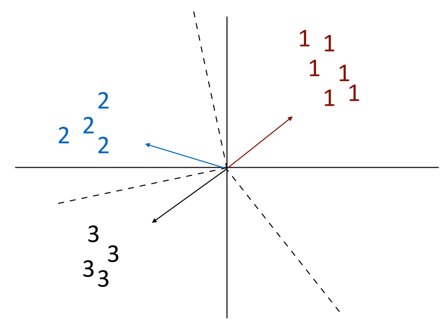
- Can separate 1 from 2+3 and 2 from 1+3 but not 3 from the others (with these features)

Multiclass Classification

- Binary classification: one weight vector defines both classes



- Multiclass classification: different weights and/or features per class



Multiclass Classification

- Formally: instead of two labels, we have an output space $\mathcal{Y}$ containing a number of possible classes
  - Same machinery that we'll use later for exponentially large output spaces, including sequences and trees
- One weight vector per class: $\mathrm{argmax}_{y \in \mathcal{Y}} \mathbf{w}_y^\top \mathbf{f}(\mathbf{x})$
- Can also view it as a feature vector per class: $\mathrm{argmax}_{y \in \mathcal{Y}} \mathbf{w}^\top \mathbf{f}(\mathbf{x}, y)$
  - Multiple feature vectors, one weight vector

features depend on choice of label now! note: this isn't the gold label

Different Weights vs. Different Features

- Different weights: $\mathrm{argmax}_{y \in \mathcal{Y}} \mathbf{w}_y^\top \mathbf{f}(\mathbf{x})$
  - Generalizes to neural networks: *f*(*x*) is the first *n*-1 layers of the network, then you apply a final linear layer at the end
- Different features: $\mathrm{argmax}_{y \in \mathcal{Y}} \mathbf{w}^\top \mathbf{f}(\mathbf{x}, y)$
  - Suppose $\mathcal{Y}$ is a structured label space (part-of-speech tags for each word in a sentence). *f*(*x*,*y*) extracts features over shared parts of these
- For linear multiclass classification with discrete classes, these are identical

Feature Extraction: Multiclass, Token Tagging Tasks

Multiclass Bag-of-words

- Decision rule: $\mathrm{argmax}_{y \in \mathcal{Y}} \mathbf{w}_y^\top \mathbf{f}(\mathbf{x})$

too many drug trials, too few patients → Health / Sports / Science

- Feature extraction:

$\mathbf{f}(\mathbf{x})$ = I[contains *drug*], I[contains *patients*], I[contains *baseball*] = [1, 1, 0]

w_{health} = [+2.1, +2.3, -5] w_{sports} = [-2.1, -3.8, +5.2]

w_{science} = [+1.1, -1.7, -1.3]

$\mathbf{w}_y^\top \mathbf{f}(\mathbf{x})$ = Health: +4.4   Sports: -5.9   Science: -0.6

argmax

Features for Tagging Tasks

DT NN VBZ DT NNS
the router blocks the packets

- Part-of-speech tagging (discussed later in the semester): make a classification decision about each word. Is "*blocks*" a verb or a noun? (~10-40 POS tags depending on the tagset, language, etc.)
- Input: sequence of words **x**, output is a sequence of tags **y**
- Simpler version: input is a sequence of words **x** and one index *i* we care about, output is the tag *y* for that position

P(*y* = *VBZ* | **x** = *the router blocks the packets*, *i* = 2)

the router *blocks* *the …*
NNS
VBZ
NN
DT
…

Features for Tagging Tasks

DT NN VBZ DT NNS
the router blocks the packets

- Do bag-of-words features work here?

[contains *the*]	[contains *router*]	[contains *is*]	[contains *packets*]	…
index 0	index 1	index 2	index 3	
f(***x***) = [1	1	0	1	…

- Every word in the sequence gets the same features — so everything gets the same label?
- Instead we need **position-sensitive features**. Let's see how this works with *different features*

Feature Extraction

DT NN VBZ DT NNS
the router blocks the packets
i = *0* *1* *2* *3* *4*

- Position-sensitive feature extractor: function from (sentence, position) => sparse feature vector describing that position in the sentence
  - "Current word": what is the word at this index?
  - "Previous word": what is the word that precedes the index?

	[currWord=*router*]	[currWord=*blocks*]	[prevWord = *router*]	
f(*x, i*=2) = [	0	1	1	…

- Feature vector only has 2 nonzero entries out of 10k+ possible
- All features coexist in the same space! Other feats (char level, …) possible

Different Features for Multiclass

- Classify *blocks* as one of 36 POS tags
- Example is a (sentence, index) pair (*x*,*i*=2): the word *blocks* in this sentence. Let's look at the **different features** view of extraction
- Different features: conjoin feats with pred label:

f(***x***, *y*=VBZ) = I[curr_word=*blocks* & tag = VBZ],
I[prev_word=*router* & tag = VBZ]
I[next_word=*the* & tag = VBZ]
I[curr_suffix=*s* & tag = VBZ]

not saying that *the* is tagged as VBZ! saying that *the* follows the VBZ word

- Get features for all tags, score, take the highest scoring one — but just one weight vector!

the router *blocks* *the packets*
NNS
VBZ
NN
DT
…

Multiclass Logistic Regression

Multiclass Logistic Regression

$$P_{\mathbf{w}}(y = \hat{y} \mid \mathbf{x}) = \frac{\exp\left(\mathbf{w}_{\hat{y}}^\top \mathbf{f}(\mathbf{x})\right)}{\sum_{y'} \exp\left(\mathbf{w}_{y'}^\top \mathbf{f}(\mathbf{x})\right)}$$

sum over output space to normalize

- Compare to binary:

$$P_{\mathbf{w}}(y = + \mid \mathbf{x}) = \frac{\exp(\mathbf{w}^\top \mathbf{f}(\mathbf{x}))}{1 + \exp(\mathbf{w}^\top \mathbf{f}(\mathbf{x}))}$$

negative class implicitly has a weight vector of all zeroes

- exp/sum(exp): also called *softmax*
- Training: maximize $\mathcal{L}(D) = \sum_{i=1}^{n} \log P_{\mathbf{w}}(y^{(i)} \mid \mathbf{x}^{(i)})$

(we'll minimize the negation of this objective)

$$= \sum_{i=1}^{n} \left(\mathbf{w}_{y^{(i)}}^\top \mathbf{f}(\mathbf{x}^{(i)}) - \log \sum_{y'} \exp(\mathbf{w}_{y'}^\top \mathbf{f}(\mathbf{x}^{(i)})) \right)$$

Training

- Multiclass logistic regression $P_{\mathbf{w}}(y = \hat{y} \mid \mathbf{x}) = \frac{\exp\left(\mathbf{w}_{\hat{y}}^\top \mathbf{f}(\mathbf{x})\right)}{\sum_{y'} \exp\left(\mathbf{w}_{y'}^\top \mathbf{f}(\mathbf{x})\right)}$
- Log loss:

$$\mathcal{L}(\mathbf{x}^{(i)}, y^{(i)}) = -\mathbf{w}_{y^{(i)}}^\top \mathbf{f}(\mathbf{x}^{(i)}) + \log \sum_{y'} \exp(\mathbf{w}_{y'}^\top \mathbf{f}(\mathbf{x}^{(i)}))$$

$$\frac{\partial}{\partial \mathbf{w}_{y^{(i)}}} \mathcal{L}(\mathbf{x}^{(i)}, y^{(i)}) = -\mathbf{f}(\mathbf{x}^{(i)}) + \frac{\mathbf{f}(\mathbf{x}^{(i)}) \exp(\mathbf{w}_{y^{(i)}}^\top \mathbf{f}(\mathbf{x}^{(i)}))}{\sum_{y'} \exp(\mathbf{w}_{y'}^\top \mathbf{f}(\mathbf{x}^{(i)}))}$$

$$\frac{\partial}{\partial \mathbf{w}_{y^{(i)}}} \mathcal{L}(\mathbf{x}^{(i)}, y^{(i)}) = -\mathbf{f}(\mathbf{x}^{(i)}) + \mathbf{f}(\mathbf{x}^{(i)}) P_{\mathbf{w}}(y^{(i)} \mid \mathbf{x}^{(i)})$$

- Update for other classes is the same but without the first term

Training

$$\frac{\partial}{\partial \mathbf{w}_{y^{(i)}}} \mathcal{L}(\mathbf{x}^{(i)}, y^{(i)}) = -\mathbf{f}(\mathbf{x}^{(i)}) + \mathbf{f}(\mathbf{x}^{(i)}) P_{\mathbf{w}}(y^{(i)} \mid \mathbf{x}^{(i)})$$

too many drug trials, too few patients

y^* = Health

$\mathbf{f}(\mathbf{x}) = [1, 1, 0]$ $P_w(y|x)$ = [0.2, 0.5, 0.3]
(made up values)

gradient $\mathbf{w}_{\text{Health}}$ = -[1, 1, 0] + 0.2[1, 1, 0]

gradient $\mathbf{w}_{\text{Sports}}$ = 0.5[1, 1, 0]

gradient $\mathbf{w}_{\text{Science}}$ = 0.3[1, 1, 0]

When we make these updates: make Sports and Science look less like the example, make Health look more like it

Multiclass Logistic Regression: Summary

- Model: $P_{\mathbf{w}}(y = \hat{y} \mid \mathbf{x}) = \dfrac{\exp\left(\mathbf{w}_{\hat{y}}^\top \mathbf{f}(\mathbf{x})\right)}{\sum_{y'} \exp\left(\mathbf{w}_{y'}^\top \mathbf{f}(\mathbf{x})\right)}$
- Inference: $\text{argmax}_{y \in \mathcal{Y}} \mathbf{w}_y^\top \mathbf{f}(\mathbf{x})$ (equivalent to finding most likely y)
- Learning: gradient descent on the log loss

$$\frac{\partial}{\partial \mathbf{w}_{y^{(i)}}} \mathcal{L}(\mathbf{x}^{(i)}, y^{(i)}) = \mathbf{f}(\mathbf{x}^{(i)})(P_{\mathbf{w}}(y^{(i)} \mid \mathbf{x}^{(i)}) - 1)$$

$$\frac{\partial}{\partial \mathbf{w}_{\tilde{y}}} \mathcal{L}(\mathbf{x}^{(i)}, y^{(i)}) = \mathbf{f}(\mathbf{x}^{(i)}) P_{\mathbf{w}}(y^{(i)} \mid \mathbf{x}^{(i)})$$

"move towards **f**(**x**) in proportion to how wrong you were"

Generative vs. Discriminative Models

Learning in Probabilistic Models

- So far we have talked about discriminative classifiers (e.g., logistic regression which models P(*y*|*x*))
- Cannot analytically compute optimal weights for such models, need to use gradient descent
- What about generative models? Let's briefly look at a generative classifier (naive Bayes) which will introduce useful concepts about maximum likelihood estimation

Naive Bayes

- Data point $x = (x_1, ..., x_n)$, label $y \in \{0, 1\}$
- Formulate a probabilistic model that places a distribution $P(x, y)$
- Compute $P(y|x)$, predict $\text{argmax}_y P(y|x)$ to classify

$$P(y|x) = \frac{P(y)P(x|y)}{P(x)}$$

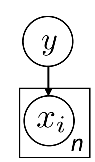
Bayes' Rule

constant: irrelevant for finding the max

$$\propto P(y)P(x|y)$$

"Naive" assumption:

$$= P(y) \prod_{i=1}^{n} P(x_i|y)$$



Maximum Likelihood Estimation

- Data points (x_j, y_j) provided (*j* indexes over examples)
- Find values of $P(y),\ \ P(x_i|y)$ that maximize data likelihood (generative):

$$\prod_{j=1}^{m} P(y_j, x_j) = \prod_{j=1}^{m} P(y_j) \left[\prod_{i=1}^{n} P(x_{ji}|y_j) \right]$$

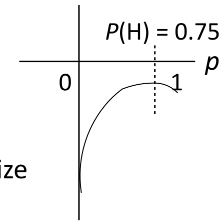
data points (*j*) features (*i*) *i*th feature of *j*th example

Maximum Likelihood Estimation

- Imagine a coin flip which is heads with probability *p*
- Observe (H, H, H, T) and maximize likelihood: $\prod_{j=1}^{m} P(y_j) = p^3(1-p)$
- Easier: maximize *log* likelihood

$$\sum_{j=1}^{m} \log P(y_j) = 3\log p + \log(1-p)$$

log likelihood



- Maximum likelihood parameters for binomial/ multinomial = read counts off of the data + normalize

Maximum Likelihood Estimation

- Data points (x_j, y_j) provided (*j* indexes over examples)
- Find values of $P(y),\ \ P(x_i|y)$ that maximize data likelihood (generative):

$$\prod_{j=1}^{m} P(y_j, x_j) = \prod_{j=1}^{m} P(y_j) \left[\prod_{i=1}^{n} P(x_{ji}|y_j) \right]$$

data points (*j*) features (*i*) *i*th feature of *j*th example

- Equivalent to maximizing log of data likelihood:

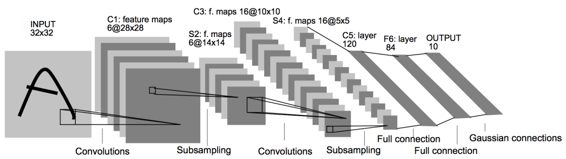
$$\sum_{j=1}^{m} \log P(y_j, x_j) = \sum_{j=1}^{m} \left[\log P(y_j) + \sum_{i=1}^{n} \log P(x_{ji}|y_j) \right]$$

- Can do this by counting and normalizing distributions!

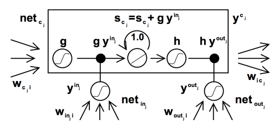
Neural Net History

History: NN "dark ages"

- Convnets: applied to MNIST by LeCun in 1998



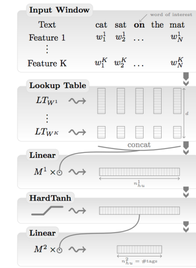
- LSTMs: Hochreiter and Schmidhuber (1997)



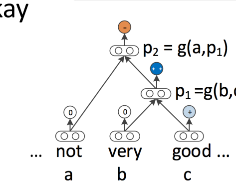
- Henderson (2003): neural shift-reduce parser, not SOTA

2008-2013: A glimmer of light…

- Collobert and Weston 2011: "NLP (almost) from scratch"
  - Feedforward neural nets induce features for sequential CRFs ("neural CRF")
  - Basically tied SOTA in 2011, but with lots of computation (two weeks of training embeddings)



- Socher 2011-2014: tree-structured RNNs working okay



- Krizhevskey et al. (2012): AlexNet for vision

2014: Stuff starts working

- Kim (2014) + Kalchbrenner et al. (2014): sentence classification / sentiment (convnets)
- Sutskever et al. + Bahdanau et al.: seq2seq for neural MT (LSTMs)
- Chen and Manning transition-based dependency parser (based on feedforward networks)
- What made these work? **Data**, **optimization** (initialization, adaptive optimizers), **representation** (good word embeddings)

Takeaways

- Two views of multiclass logistic regression:
  - Different weights: one weight vector per class, fixed features
  - Different features: single weight vector for all classes, features differ for each class (but in a systematic way)
- Gradient looks like binary logistic regression gradient: softly move gold weight vector towards the example (also move all other weight vectors away from the example)
- Next time: neural networks
  - Extension of multiclass logistic regression with a nonlinearity