

CS4705

Supervised Machine Learning

Some of the slides in this class were adapted from
The Oxford slides and from slides from Dragomir
Radev

Announcements and Questions

- If you were on the waitlist and moved on the class, you have 5 days for HW0 from the time you joined the waitlist. To get a google coupon, send email to:
- If you were not on the waitlist, you have until Sat 2:40 pm(which would use up all late days) to turn it in or you will be dropped from the class.
- I will be away at a workshop Wed-Fri this week. No office hours Wed. Jessica Ouyang will be teaching Thurs.
- Questions?

Outline for Today's Class

- Perplexity and smoothing
- Text Classification: definition
- Bag of words and feature vectors
- Generative vs discriminative classifiers
- Discriminative classifiers
- Optimization
- Loss
- Regularization
- HW1

Some Important Concepts

- **Smoothing and Backoff** : how do you handle unseen n-grams?
- **Perplexity and entropy**: how do you estimate how well your language model fits a corpus once you're done?

Perplexity

- Information theoretic metric
 - Useful in measuring how well a **grammar** or **language model (LM)** models a natural language or a corpus

Perplexity: A function of the probability that a language model assigns to the test corpus.

$$P(W) = P(w_1, w_2 \dots w_N)^{-1/N}$$

What perplexity does a LM(1) assign to the sentences of a test corpus, compared to another LM(2)?

Some Useful Empirical Observations

- A small number of events occur with high frequency
- A large number of events occur with low frequency
- You can quickly collect statistics on the high frequency events
- You might have to wait an arbitrarily long time to get valid statistics on low frequency events
- Some of the zeroes in the table are really zeros
But others are simply low frequency events you haven't seen yet. How to address?

Smoothing

- Words follow a Zipfian distribution
 - Small number of words occur very frequently
 - A large number are seen only once
 - Zipf's law: a word's frequency is approximately inversely proportional to its rank in the word distribution list
- Zero probabilities on one bigram cause a zero probability on the entire sentence

Smoothing is like Robin Hood: Steal from the rich and give to the poor (in probability mass)

- We often want to make predictions from sparse statistics:

$P(w \mid \text{denied the})$

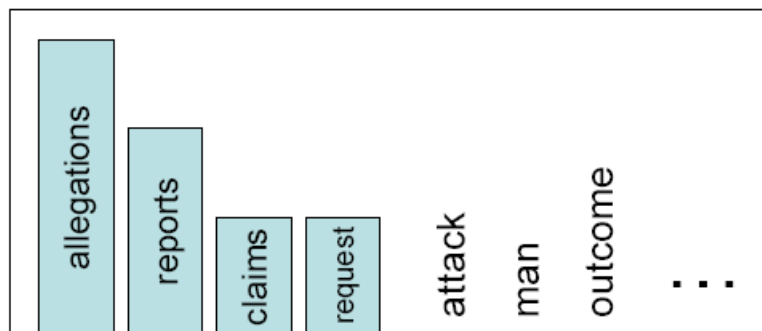
3 allegations

2 reports

1 claims

1 request

7 total



- Smoothing flattens spiky distributions so they generalize better

$P(w \mid \text{denied the})$

2.5 allegations

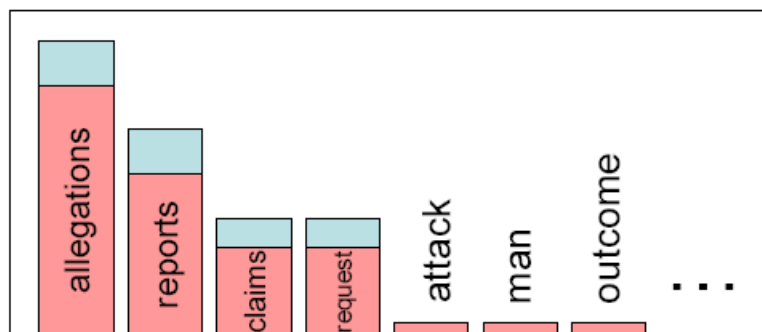
1.5 reports

0.5 claims

0.5 request

2 other

7 total



- Very important all over NLP, but easy to do badly!

Smoothing Methods

- Add-one smoothing (easy, but inaccurate)
 - Add 1 to every word count (Note: this is type)
 - Increment normalization factor by Vocabulary size: N (tokens) + V (types) :

$$p_i^* = \frac{c_i + 1}{N + V}$$

- Backoff models
 - When a count for an n -gram is 0, back off to the count for the $(n-1)$ -gram
 - These can be weighted
- Class-based smoothing
 - For certain types of n -grams, back off to the count of its syntactic class
 - E.g., Count ProperNouns in place of names (e.g., Obama)
- Good-Turing
 - Re-estimate amount of probability mass for zero (or low count) ngrams by looking at ngrams with higher counts
 - Estimate

$$c^* = (c + 1) \frac{N_{c+1}}{N_c}$$

Text Classification

- Authorship
- German vs English
- Genre
- Positive vs negative reviews

Genre

- Devastating, but yet amazing storm. IMBY just some branches, ton of leaves, ...This surpasses any Big daddy storm, should be called Big Mama. (Computer Guy)
- Produced by a team of 26 scientists led by the University of New South Wales Climate Research Centre, the Diagnosis convincingly proves that the effects of global warming have gotten worse in the last three years. (Somerville et al 2011)
- Hurricane Sandy churned about 290 miles off the Mid-Atlantic coast Sunday night, with the National Hurricane Center reporting that the monster storm was expected to come ashore with near-hurricane-force winds and potentially "life-threatening" storm surge flooding.
- *It was broad day—eight or nine o'clock; the storm raging, in lieu of the batteries; and someone knocking and calling at my door. 'What is the matter?' I cried. 'A wreck! Close by!' I sprung out of bed, and asked, what wreck? 'A schooner, from Spain or Portugal, laden with fruit and wine. Make haste, sir, if you want to see her! ...'*

Features

- *What words give a clue about genre?*
- *What other information could we use?*

Response

Positive or Negative Review

- I **hate** the **iphone** headphones they hurt **my** ears
- Well I absolutely **love my iPhone** 6+. Battery life is just awesome. Having 40% left at the end of the day is great.
- Received email from Voldemort this morning & he has demanded that 10pgs of this grant be finished my Monday.
- Do you love what you do for life? I do and have fun:-)

Features

- *What words give a clue about sentiment?*
- *What other information could we use?*

Response

Other Classification Tasks in NLP

- POS tagging
- Parsing
- Word sense disambiguation
- Demographics in social media (e.g., gender)
- Named entity detection

Types of Classification

- Binary (e.g., adult? T,F)
- Multi-class (e.g., age: child, teen, adult)
- Multi-label (e.g., child, American)
- Result of clustering (no labels)

Generative vs Discriminative Classifiers

- Generative
 - Probabilistic
 - Specify a joint probability distribution over observations and targets: $P(c,d)$
 - Bayes rule enables a conditional distribution
- Discriminative
 - Provide a model for the target variable
 - Use analysis of observed variables
 - Learn boundaries between classes
 - Infer outputs based on inputs: $P(c/d)$

Generative vs Discriminative Classifiers

- Generative
 - Probabilistic
 - Specify a joint probability distribution over observations and targets: $P(c,d)$
 - Bayes rule enables a conditional distribution
- ***Discriminative***
 - Provide a model for the target variable
 - Use analysis of observed variables
 - Learn boundaries between classes
 - Infer outputs based on inputs: $P(c/d)$

Bag of Words Classifier

- Vector to represent all possible words
 - Value = number of times word appears in the document
 - Unigrams
- Vector or matrix to represent all possible pairs of words
 - Value = number of times bigram appears in the document
 - Bigrams
- Bag because we've thrown out information about order, syntax, sentence boundaries, paragraphs

Feature Vector Representation

- Unigrams
- Bigrams
- *How big would your vector be?*
- Adding meta-level features
 - Document length, sentence length, sentence position, word position, author name

Response

Bag of words representation

- Devastating, but yet amazing storm. IMBY just some branches, ton of leaves, ...This surpasses any Big daddy storm, should be called Big Mama. (Computer Guy)

| | |
|--------------|---|
| be | 1 |
| big | 2 |
| branches | 1 |
| batteries | 0 |
| convincingly | 0 |
| computer | 1 |
| daddy | 1 |
| day | 0 |
| devastating | 1 |
| fruit | 0 |

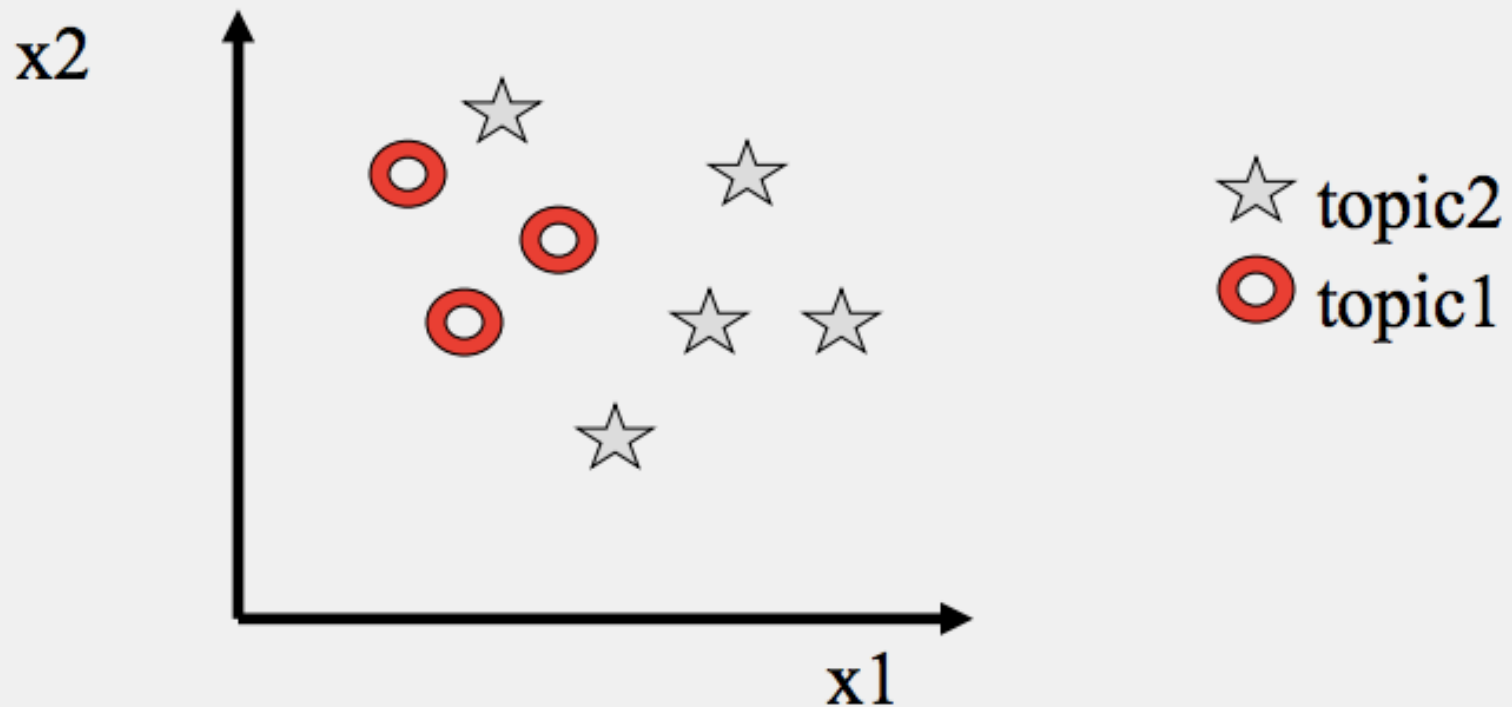
Weights

- Positive vs negative reviews
 - Surprising vs wonderful vs awful
- Genre: twitter vs novels vs scientific journal articles
 - Exclaimed vs neuro-transmitter vs LOL

Learning weights from data

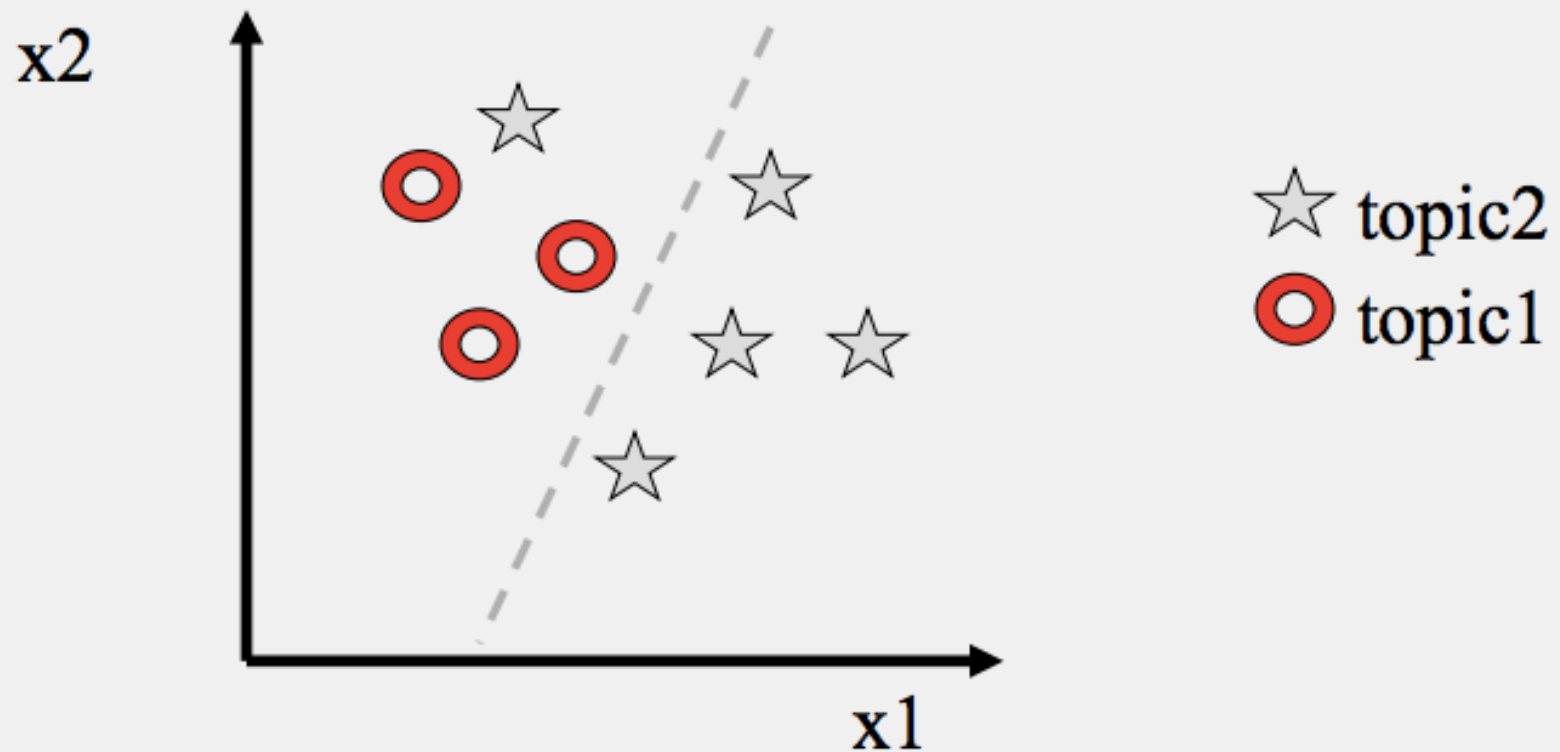
- Training data
 - Labeled text
 - Sentences or paragraphs drawn from online sources
 - Reddit
 - Novels
 - Journals
 - News
 - Feature vector: bag of words plus output prediction

Vector Space Classification



Slide from Dragomir Radev

Linear boundary




Slide from Dragomir Radev

Feature Vector Representation

| | | |
|----------|--------------------|---|
| x_1 | be | 1 |
| x_2 | big | 2 |
| x_3 | branches | 1 |
| x_4 | batteries | 0 |
| x_5 | convincingly | 0 |
| x_6 | computer | 1 |
| x_7 | daddy | 1 |
| x_8 | day | 0 |
| x_9 | devastating | 1 |
| x_{10} | fruit | 0 |
| Y_1 | Social media | 1 |
| Y_2 | Novel | 0 |
| Y_3 | Scientific journal | 0 |
| Y_4 | News | 0 |

- Devastating, but yet amazing storm. IMBY just some branches, ton of leaves, ... This surpasses any Big daddy storm, should be called Big Mama.
(Computer Guy)



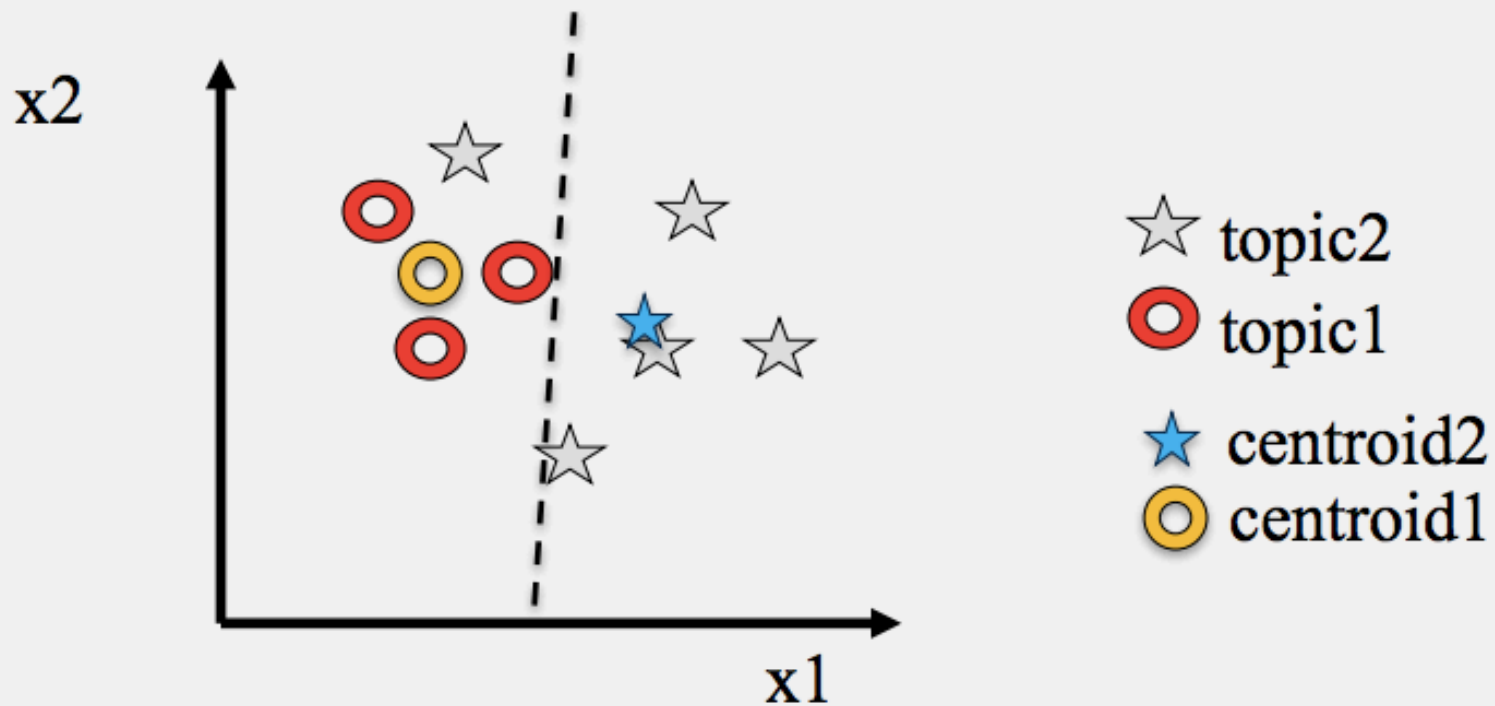
One hot vector for output class

Classification using centroids

- Centroid
 - The point most representative of a class
- Compute centroid by finding vector average of known class members
 - *How would we do this?*
- Decision boundary is a line that is equidistant from the two centroids
- New document on one side of the line goes in class 1 and on other side goes in class2

Response

Classification Using Centroids



Linear Separators

$$f(x) = \Theta X + b$$

where

Θ is a vector of weights: $w_1 \dots w_n$

X is the input vector

b is a constant

Two dimensional space:

$$w_1 x_1 + w_2 x_2 = b$$

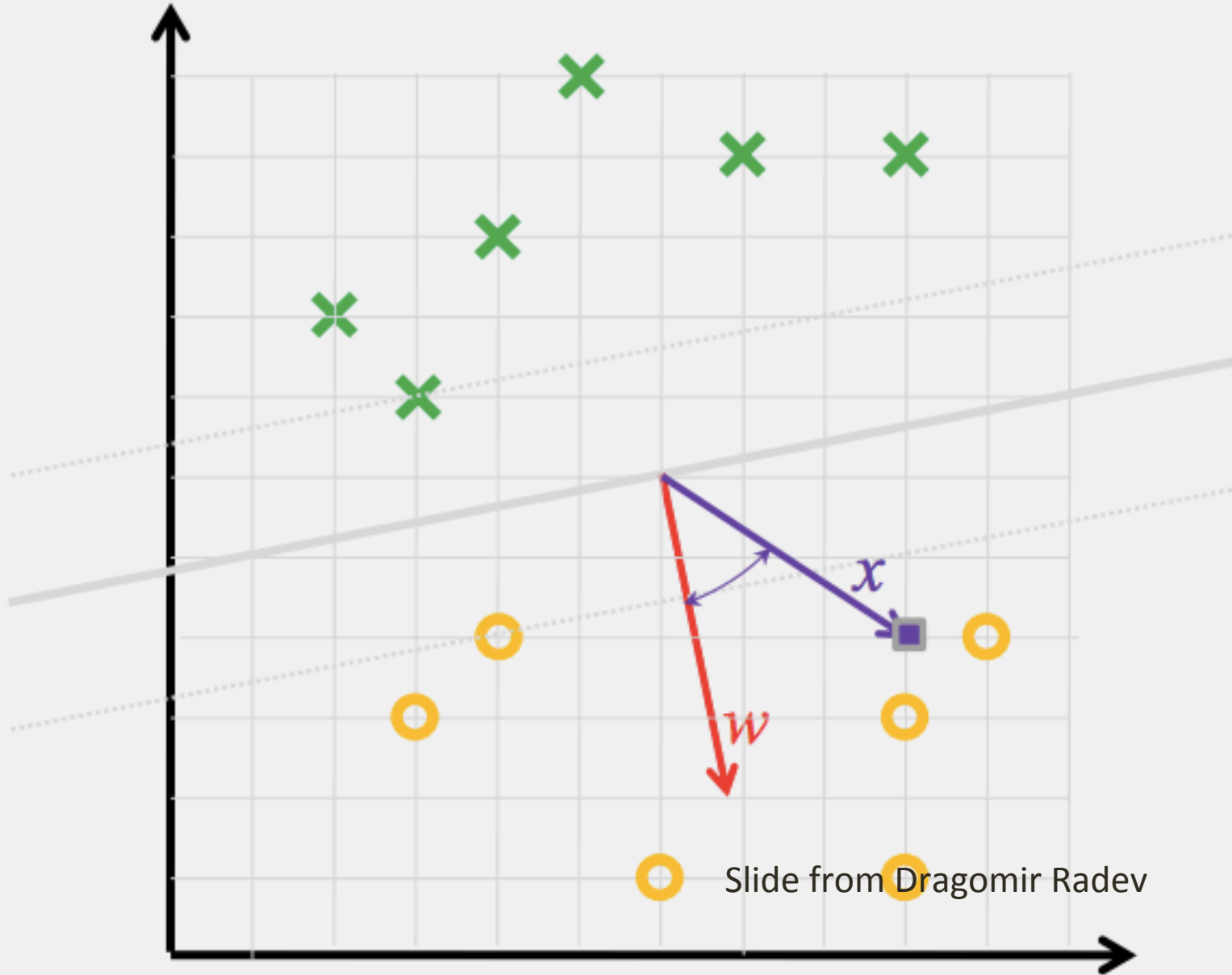
In n-dimensional spaces:

$$\Theta X = \sum_{i=1}^n w_i x_i = w_1 x_1 + w_2 x_2 + \dots + w_n x_n$$

One can also add $w_0 = 1$, $x_0 = b$ to account for bias

Pass output of $f(x)$ to the sign function, mapping negative values to -1 and positive values to 1

Decision Boundary



Slide from Dragomir Radev

Example

$$\Theta X = b$$

Assume $b = 0$

| Social mediaany | .2 | any | -.2 |
|-----------------|----|--------------|-----|
| be | .2 | be | -.2 |
| Big | 1 | Big | -.5 |
| surpasses | .5 | surpasses | -.7 |
| computer | .3 | batteries | -.8 |
| daddy | 1 | convincingly | -.1 |
| day | .7 | day | -.8 |
| fruit | .5 | fruit | -.5 |
| Storm | .5 | Storm | -.5 |
| this | .3 | this | -.3 |

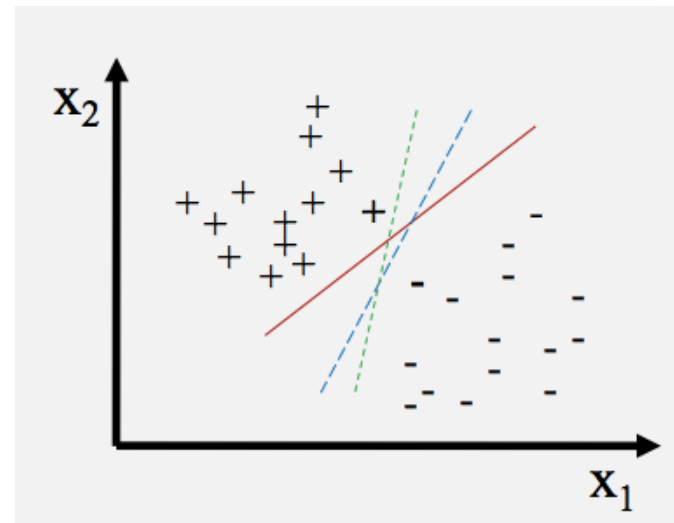
“This surpasses any Big daddy storm”

$$\Theta X = .3*1 + .5*1 + .2*1 + 1*2 + 1*1 + .5*1 = 4.5 > 0$$

How to find the linear boundary?

- How to find the vector of weights Θ ?

- Many methods
 - Perceptron
 - Linear least squares
 - Logistic regression



- Problem
 - There are infinite number of linear boundaries if the classes are linearly separated
 - Maximum margin: support vector machines

Training using optimization

- Select values for w
- Compute $f(x)$
- Compare $f(x)$ output to gold labels and compute *loss*
- Adjust w

Using a loss function

- Training data
 - $x_1 x_2 \dots x_n$ (input)
 - $y_1 y_2 \dots y_n$ (labels)
- Algorithm that returns $f(x)$ with predictions \hat{y}
- Loss function $L(\hat{y}, y)$
- Parameters of the learned function (Θ, b) set to minimize L

Loss

- Given: labeled training set, L , parameterized function $f(x, \Theta)$ where $\Theta = w_1 \dots w_n, b$
- Corpus wide loss = average loss over all training examples:

$$\mathcal{L}(\Theta) = \frac{1}{n} \sum_{i=1}^n L(f(x_i; \Theta), y_i)$$

- Training algorithm sets Θ to minimize L :

$$\hat{\Theta} = \underset{\Theta}{\operatorname{argmin}} \mathcal{L}(\Theta) = \underset{\Theta}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^n L(f(x_i; \Theta), y_i)$$

- To prevent overfitting, a regularization parameter $R(\Theta)$ is added:

$$\hat{\Theta} = \underset{\Theta}{\operatorname{argmin}} \left(\overbrace{\frac{1}{n} \sum_{i=1}^n L(f(x_i; \Theta), y_i)}^{\text{loss}} + \overbrace{\lambda R(\Theta)}^{\text{regularization}} \right)$$

Loss Functions

- Can be an arbitrary function mapping two vectors to a scalar
- Hinge Loss (binary)
 - Classifier output: \hat{y} , gold output: y (-1,1)
 - \hat{y} correct if $\hat{y} \cdot y > 1$ (Classification rule is $\text{sign}(\hat{y})$)

$$L_{\text{hinge}(\text{binary})}(\hat{y}, y) = \max(0, 1 - y \cdot \hat{y})$$

Loss is 0 when \hat{y} , y share the same sign and $|\hat{y}| \geq 1$, linear otherwise

- Attempts to achieve correct classification with a margin of ≤ 1

Hinge Loss (multi-class)

- Classifier output: $\hat{Y} = \hat{y}_1 \dots \hat{y}_n$
- Y is one-hot vector for correct class
- Prediction = $\operatorname{argmax}_i \hat{y}_i$

$$L_{\text{hinge(multiclass)}}(\hat{y}, y) = \max(0, 1 - (\hat{y}_{[t]} - \hat{y}_{[k]}))$$

- where t is the correct class and k is the highest scoring class where $k \neq t$
- Scores the correct class above all other classes with a margin of at least 1

Many other loss functions

- Log loss
- Binary cross entropy
- Categorical cross entropy
- Ranking losses

Regularization

- Consider the case where one or more documents are mis-labeled
 - Text from a novel may be mis-labeled as social media if posted as a quote
- The classifier will attempt to learn weights that promote words characteristic of novels as predictors of social media
- Overfitting can also occur when the social media documents in the training set are not representative

Two Common regularizers

- L_2 regularization
 - Keeps sum of squares of parameter values low

$$R_{L_2}(\mathbf{W}) = \|\mathbf{W}\|_2^2 = \sum_{i,j} (\mathbf{W}_{[i,j]})^2$$

- Gaussian prior or weight decay (Here W is weights not including b)
 - Prefers to decrease parameter with high weight by 1 than 10 parameters with low weights
- L_1 regularization
 - Keeps sum of absolute value of parameters low

$$R_{L_1}(\mathbf{W}) = \|\mathbf{W}\|_1 = \sum_{i,j} |\mathbf{W}_{[i,j]}|$$

Punished uniformly for high and low values

Gradient based optimization

- Repeat until $L < \text{margin}$
 - Compute L over the training set
 - Compute gradients of Θ with respect to L
 - Move the parameters in the opposite direction of the gradient

Stochastic Gradient Descent

Algorithm 1 Online Stochastic Gradient Descent Training

Input:

- Function $f(\mathbf{x}; \Theta)$ parameterized with parameters Θ .
- Training set of inputs $\mathbf{x}_1, \dots, \mathbf{x}_n$ and desired outputs y_1, \dots, y_n .
- Loss function L .

```
1: while stopping criteria not met do
2:   Sample a training example  $\mathbf{x}_i, y_i$ 
3:   Compute the loss  $L(f(\mathbf{x}_i; \Theta), y_i)$ 
4:    $\hat{\mathbf{g}} \leftarrow$  gradients of  $L(f(\mathbf{x}_i; \Theta), y_i)$  w.r.t  $\theta$ 
5:    $\Theta \leftarrow \Theta - \eta_t \hat{\mathbf{g}}$ 
6: return  $\Theta$ 
```

Problem

- Error is calculated based on just one training sample
- May not be representative of corpus wide loss
- Instead calculate the error based on a set of training examples: *minibatch*
- -> Minibatch stochastic gradient descent

Computing Gradients

$$\frac{\partial L}{\partial \mathbf{b}_{[i]}} = \begin{cases} -1 & i = t \\ 1 & i = k \\ 0 & \text{otherwise} \end{cases}$$

$$\frac{\partial L}{\partial \mathbf{W}_{[i,j]}} = \begin{cases} \frac{\partial(-\mathbf{x}_{[i]} \cdot \mathbf{W}_{[i,t]})}{\partial \mathbf{W}_{[i,t]}} & = -\mathbf{x}_{[i]} & j = t \\ \frac{\partial(\mathbf{x}_{[i]} \cdot \mathbf{W}_{[i,k]})}{\partial \mathbf{W}_{[i,k]}} & = \mathbf{x}_{[i]} & j = k \\ 0 & \text{otherwise} \end{cases}$$

Summary

- Smoothing helps to account for zero valued n-grams
- Text classification using feature vectors representing n-grams and other properties
- Discriminative learning
- Methods for optimization, loss functions and regularization

Questions?