TTIC 31190: Natural Language Processing

Kevin Gimpel

Winter 2016

Lecture 3: Words

- Assignment 1 has been posted
- Due 11:59 pm on Wednesday, January 20th

- We will start class 5 minutes late from now on, due to several students taking algorithms across campus

- My office hours are Mondays 3-4pm, #531 (or by appointment)
- TA office hours are Thursdays 4-5pm, #501

- If you're auditing, you may still turn in the homework and we will give you feedback (though we may not give your homework as much attention as others)

- If you didn't receive an email from me this details, then please email me with your name/email and let me know whether you are taking course for credit

Today

- review of loss functions and subgradients from last week (useful for homework)

- start words (more about words and lexical semantics on Thursday)

Empirical Risk Minimization with Surrogate Loss Functions

- given training data: $\mathcal{T} = \{\langle \boldsymbol{x}^{(i)}, y^{(i)} \rangle\}_{i=1}^{|\mathcal{T}|}$
  where each $y^{(i)} \in \mathcal{L}$ is a label
- we want to solve the following:

$$\hat{\boldsymbol{\theta}} = \operatorname*{argmin}_{\boldsymbol{\theta}} \sum_{i=1}^{|\mathcal{T}|} \mathrm{loss}(\boldsymbol{x}^{(i)}, y^{(i)}, \boldsymbol{\theta})$$

Empirical Risk Minimization with Surrogate Loss Functions

- given training data: $\mathcal{T} = \{\langle \boldsymbol{x}^{(i)}, y^{(i)} \rangle\}_{i=1}^{|\mathcal{T}|}$
  where each $y^{(i)} \in \mathcal{L}$ is a label
- we want to solve the following:

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \sum_{i=1}^{|\mathcal{T}|} \operatorname{loss}(\boldsymbol{x}^{(i)}, y^{(i)}, \boldsymbol{\theta})$$

many possible loss functions to consider optimizing

Cost Functions

- cost function: scores output against a gold standard

$$\text{cost} : \mathcal{L} \times \mathcal{L} \rightarrow \mathbb{R}_{\geq 0}$$

- should reflect the evaluation metric for your task

- usual convention: $\text{cost}(y, y) = 0$

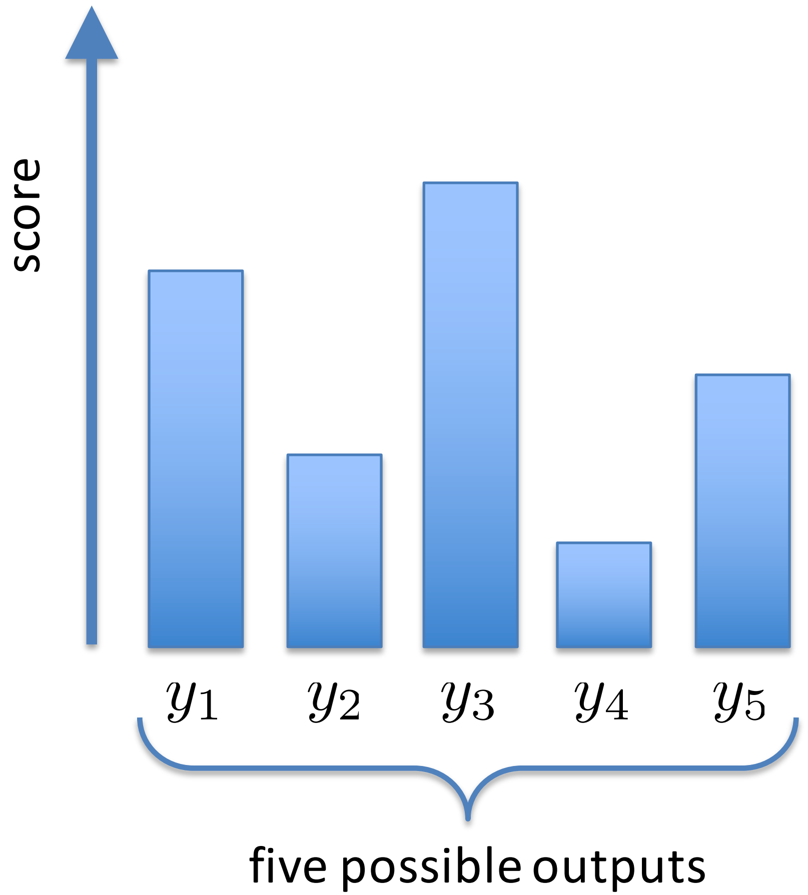
Surrogate Loss Functions

cost loss / 0-1 loss: $\text{loss}_{\text{cost}}(\boldsymbol{x}, y, \boldsymbol{\theta}) = \text{cost}(y, \text{classify}(\boldsymbol{x}, \boldsymbol{\theta}))$

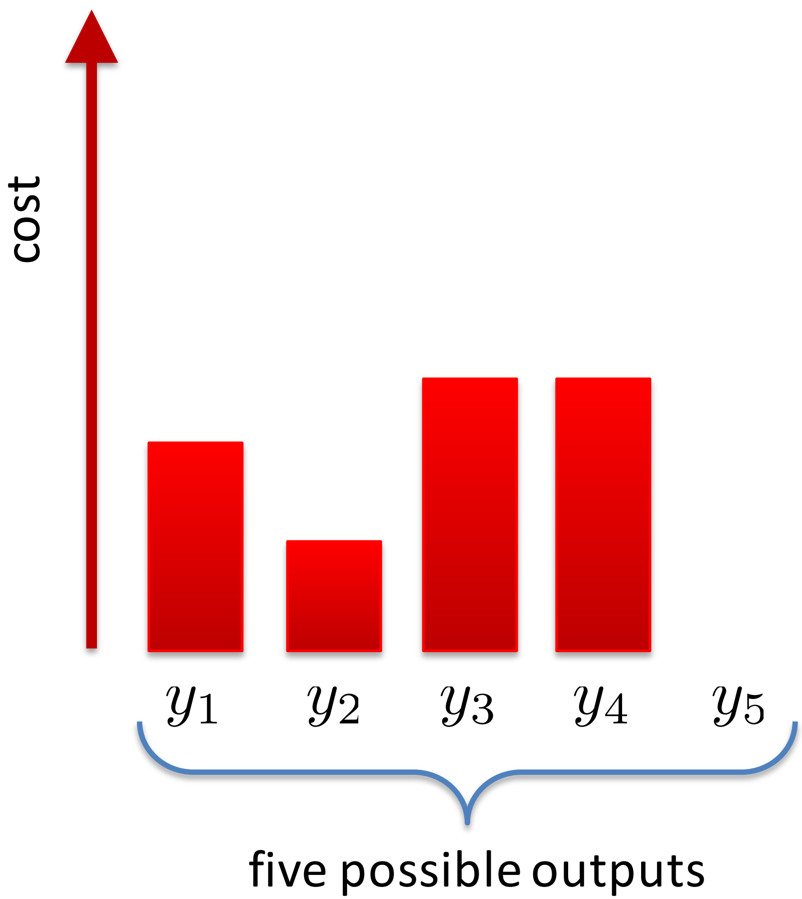
max-score loss:

$$\text{loss}_{\text{maxscore}}(\boldsymbol{x}, y, \boldsymbol{\theta}) = -\text{score}(\boldsymbol{x}, y, \boldsymbol{\theta})$$

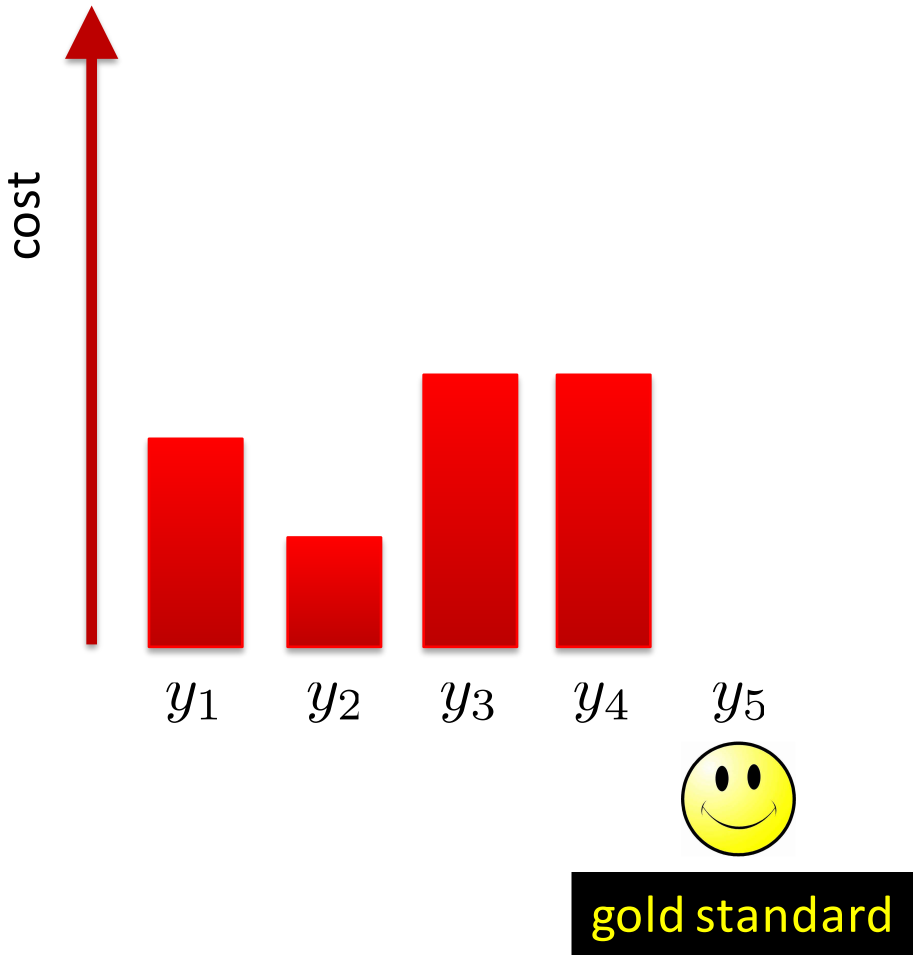
Visualization



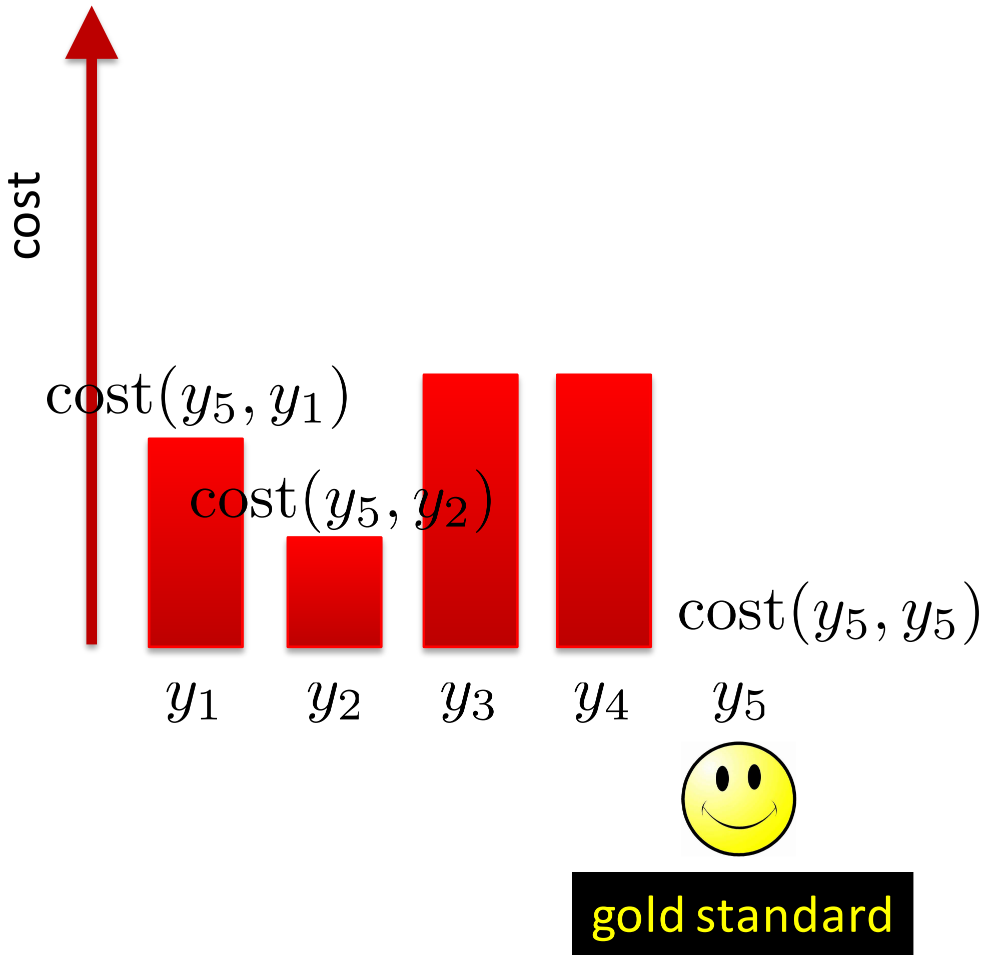
Visualization



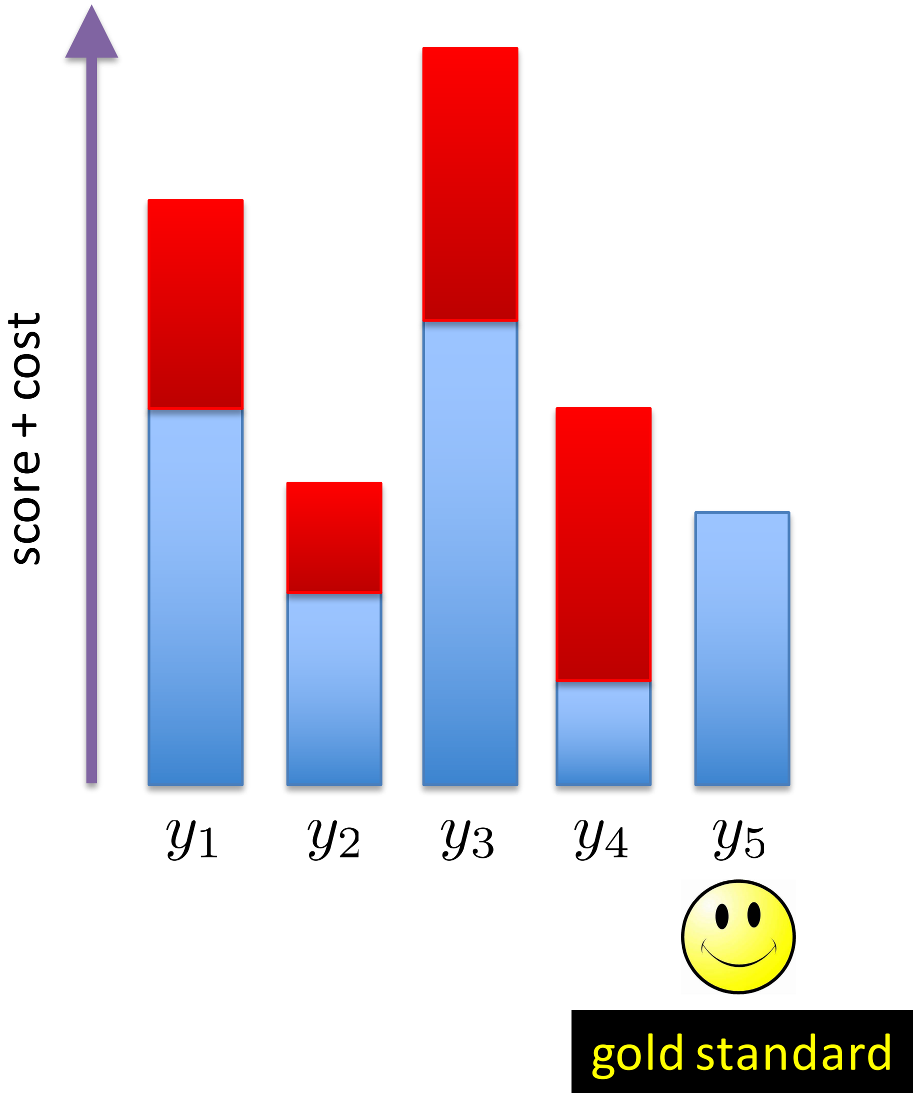
Visualization



Visualization

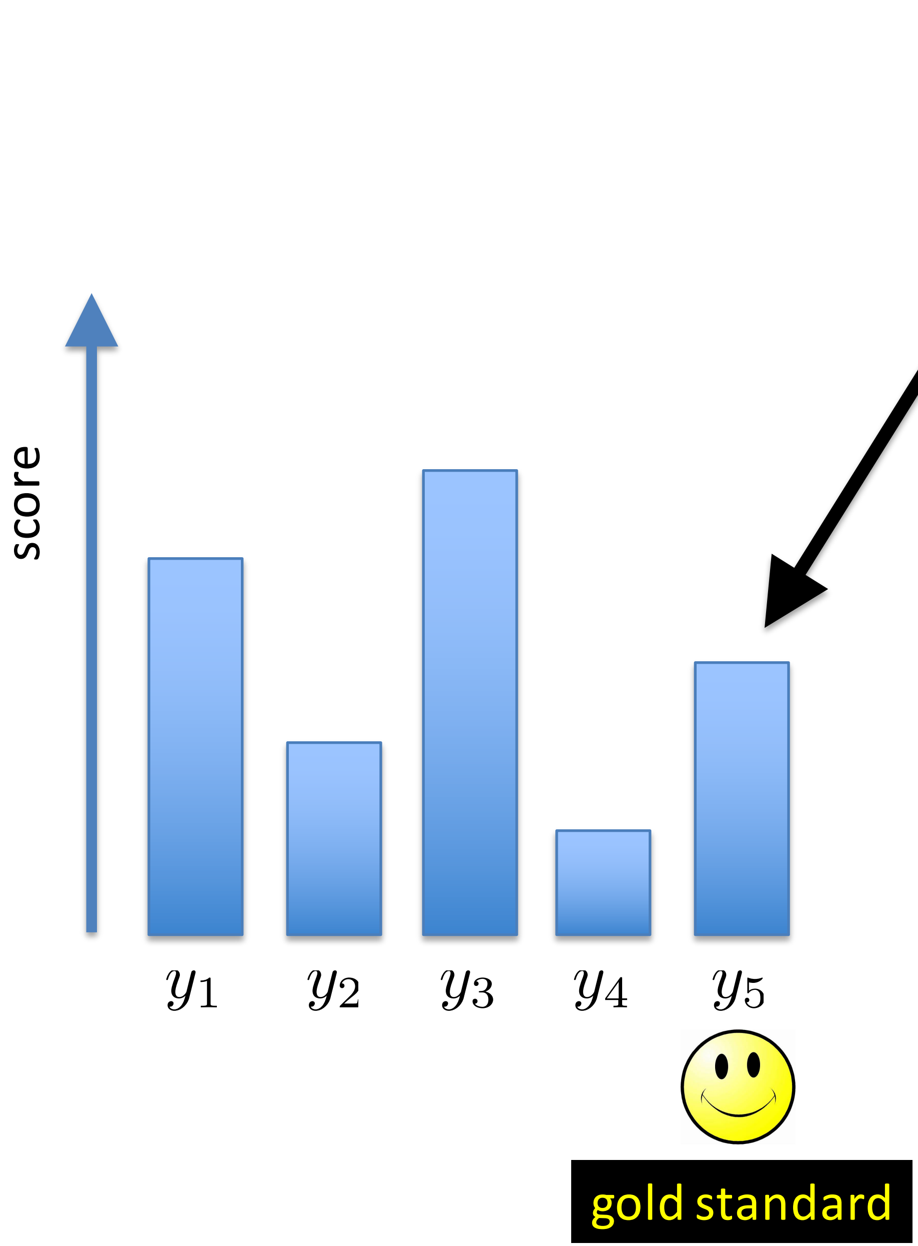


Visualization

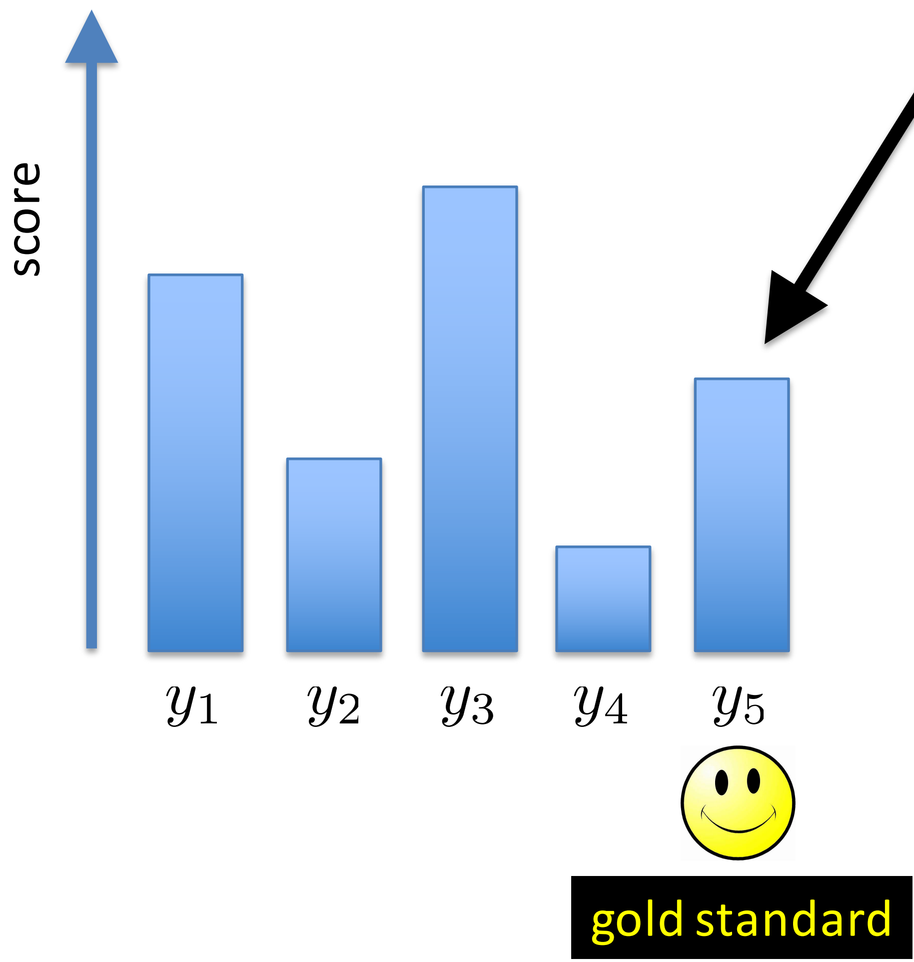


$$\text{loss}_{\text{maxscore}}(\boldsymbol{x}, y, \boldsymbol{\theta}) = -\text{score}(\boldsymbol{x}, y, \boldsymbol{\theta})$$

$$\text{loss}_{\text{maxscore}}(\boldsymbol{x}, y, \boldsymbol{\theta}) = -\text{score}(\boldsymbol{x}, y, \boldsymbol{\theta})$$

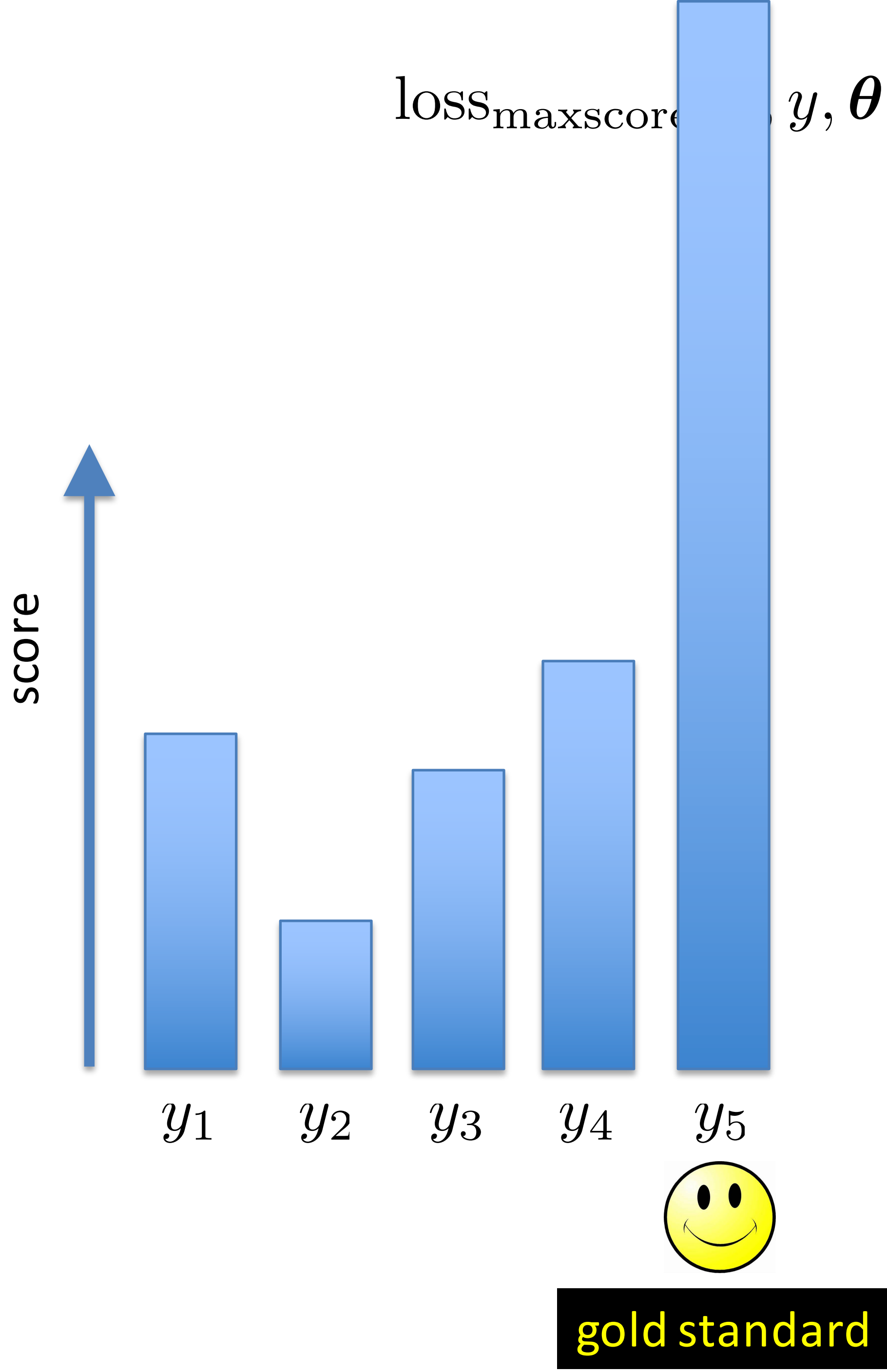


$$\text{loss}_{\text{maxscore}}(\boldsymbol{x}, y, \boldsymbol{\theta}) = -\text{score}(\boldsymbol{x}, y, \boldsymbol{\theta})$$



effect of learning?

$$\text{loss}_{\text{maxscore}}(\boldsymbol{x}, y, \boldsymbol{\theta}) = -\text{score}(\boldsymbol{x}, y, \boldsymbol{\theta})$$



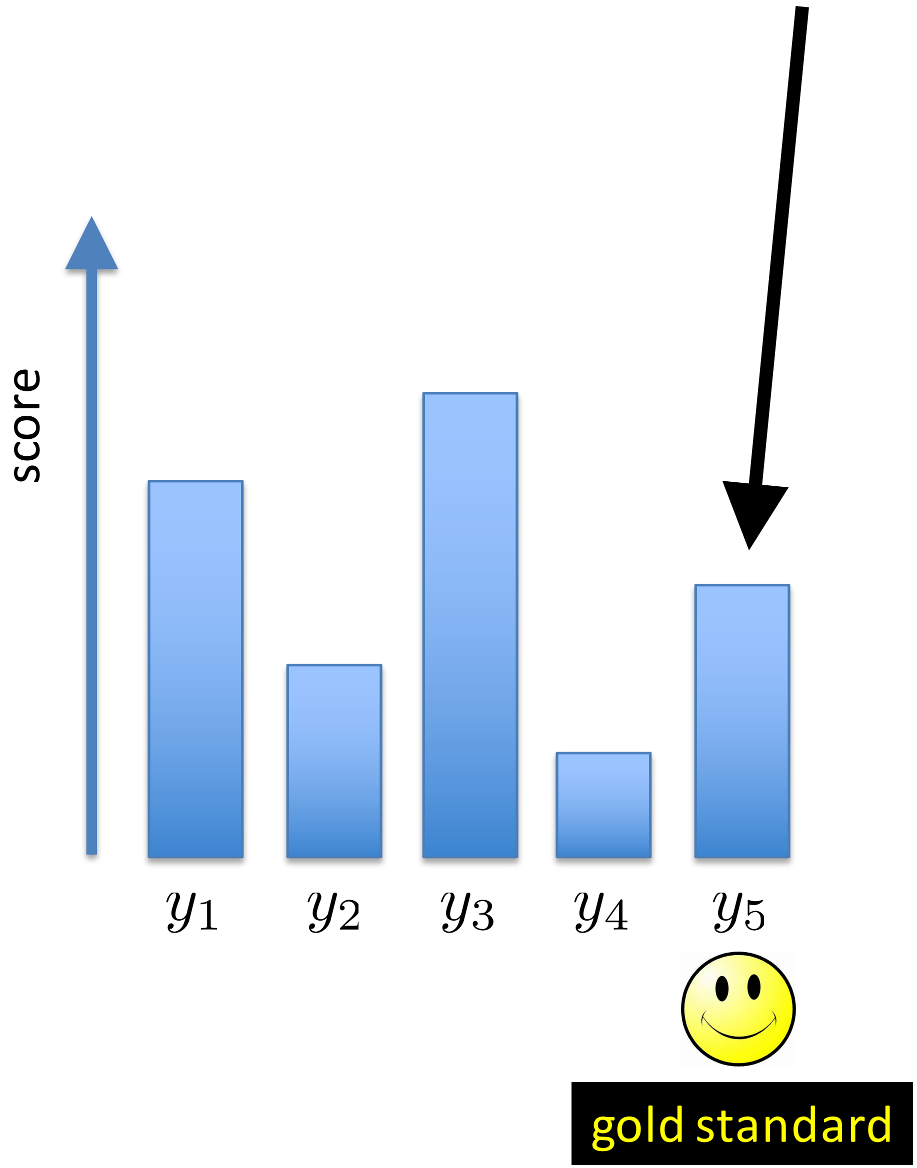
effect of learning:
score of gold standard
will go to infinity

perceptron loss:

$$\text{loss}_{\text{perc}}(\boldsymbol{x}, y, \boldsymbol{\theta}) = -\text{score}(\boldsymbol{x}, y, \boldsymbol{\theta}) + \max_{y' \in \mathcal{L}} \ \text{score}(\boldsymbol{x}, y', \boldsymbol{\theta})$$

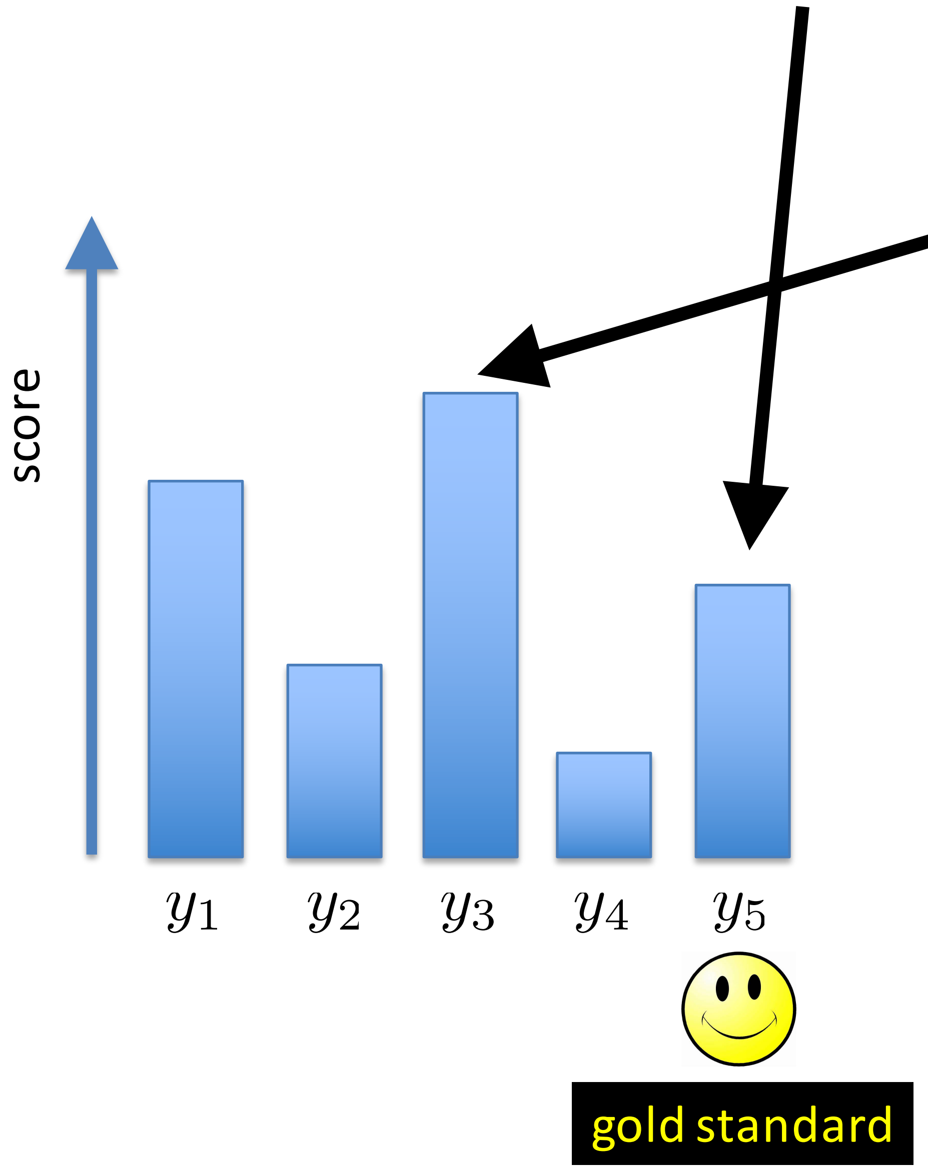
perceptron loss:

$$\text{loss}_{\text{perc}}(\boldsymbol{x}, y, \boldsymbol{\theta}) = -\text{score}(\boldsymbol{x}, y, \boldsymbol{\theta}) + \underbrace{\max_{y' \in \mathcal{L}} \ \text{score}(\boldsymbol{x}, y', \boldsymbol{\theta})}$$



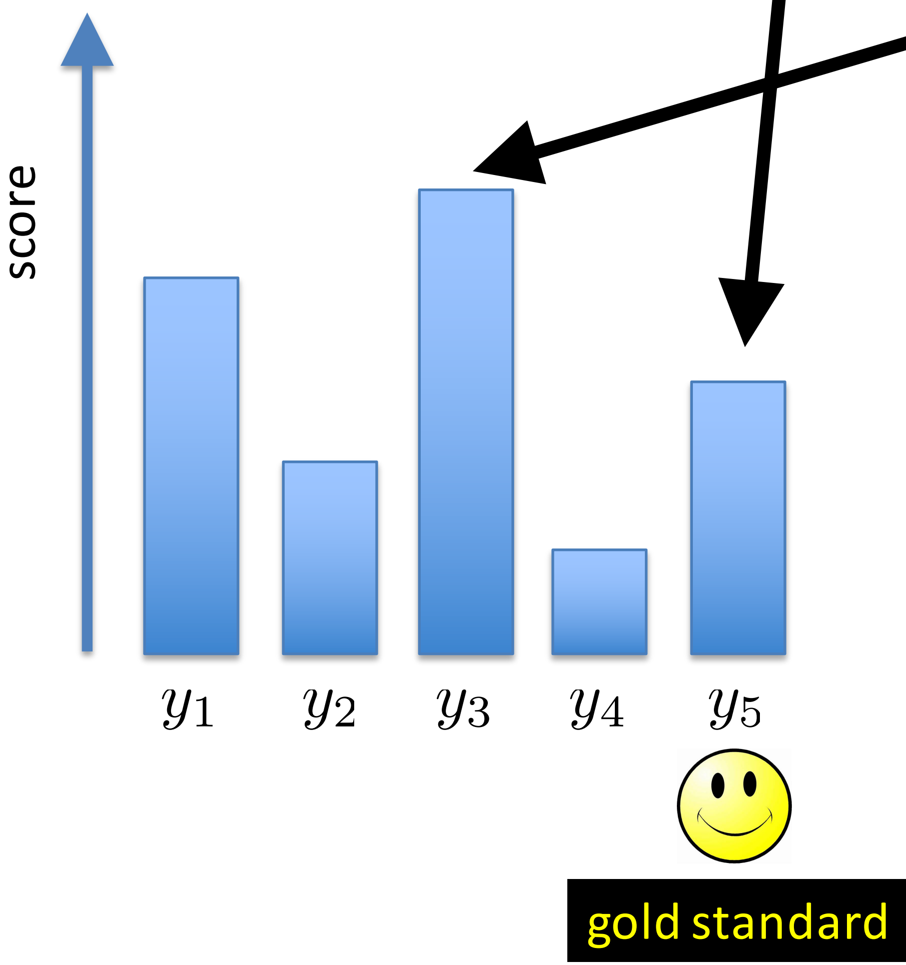
perceptron loss:

$$\mathrm{loss}_{\mathrm{perc}}(\boldsymbol{x}, y, \boldsymbol{\theta}) = -\mathrm{score}(\boldsymbol{x}, y, \boldsymbol{\theta}) + \underbrace{\max_{y' \in \mathcal{L}} \mathrm{score}(\boldsymbol{x}, y', \boldsymbol{\theta})}$$



perceptron loss:

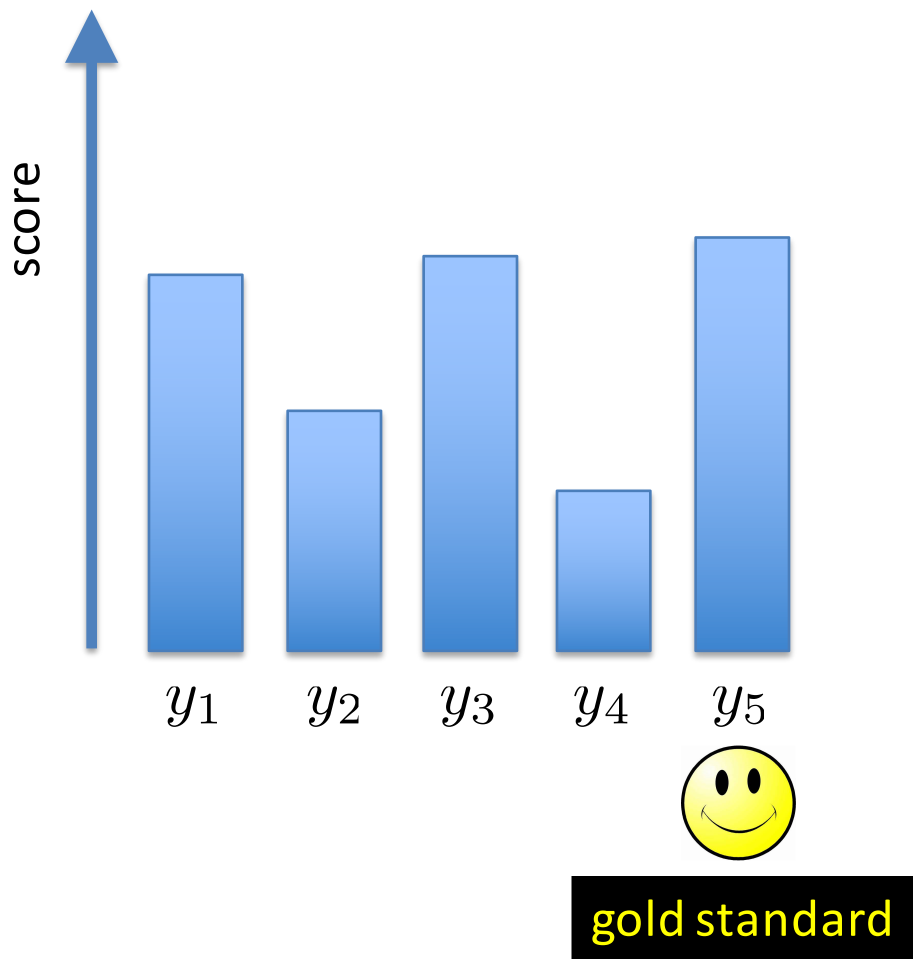
$$\text{loss}_{\text{perc}}(\boldsymbol{x}, y, \boldsymbol{\theta}) = -\text{score}(\boldsymbol{x}, y, \boldsymbol{\theta}) + \underbrace{\max_{y' \in \mathcal{L}} \text{score}(\boldsymbol{x}, y', \boldsymbol{\theta})}$$



effect of learning?

perceptron loss:

$$\text{loss}_{\text{perc}}(\boldsymbol{x}, y, \boldsymbol{\theta}) = -\text{score}(\boldsymbol{x}, y, \boldsymbol{\theta}) + \max_{y' \in \mathcal{L}} \text{score}(\boldsymbol{x}, y', \boldsymbol{\theta})$$



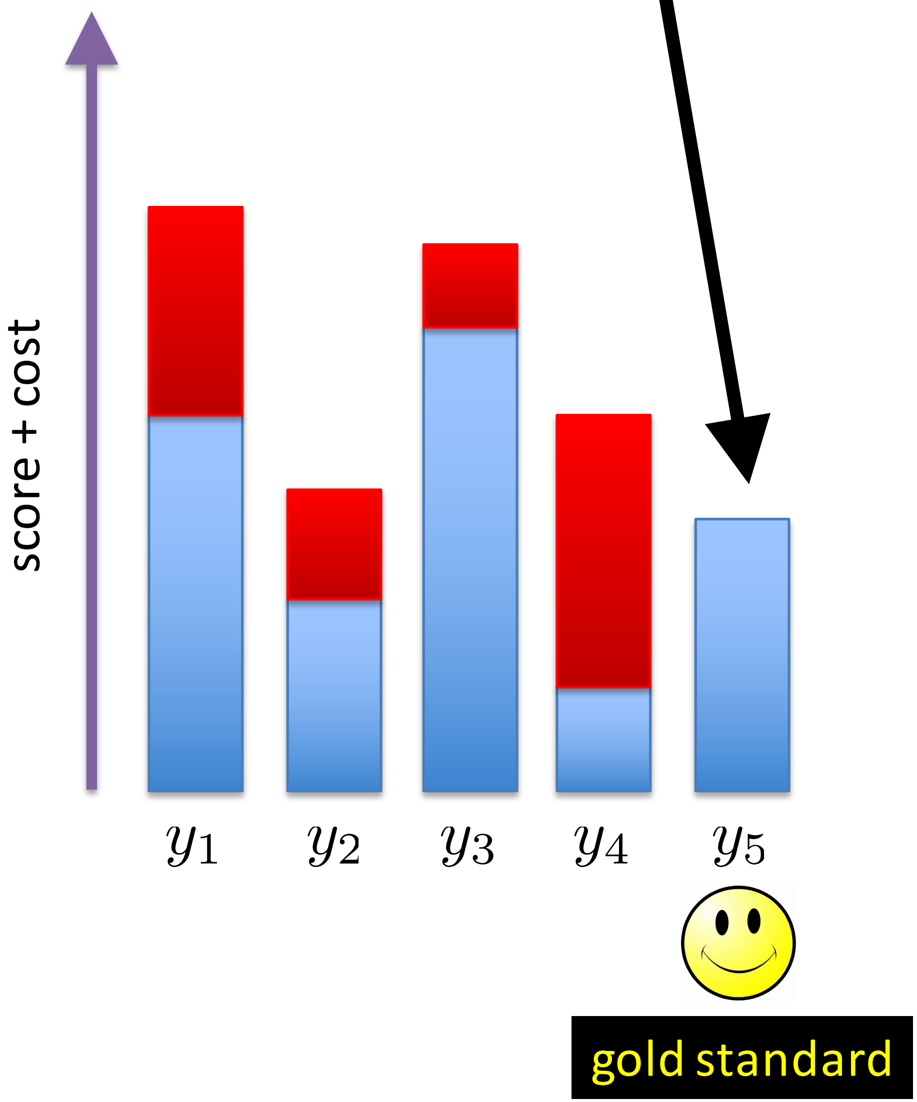
effect of learning:
gold standard will have highest score

hinge loss:

$$\text{loss}_{\text{hinge}}(\boldsymbol{x}, y, \boldsymbol{\theta}) = -\text{score}(\boldsymbol{x}, y, \boldsymbol{\theta}) + \max_{y' \in \mathcal{L}} \left(\text{score}(\boldsymbol{x}, y', \boldsymbol{\theta}) + \text{cost}(y, y')\right)$$

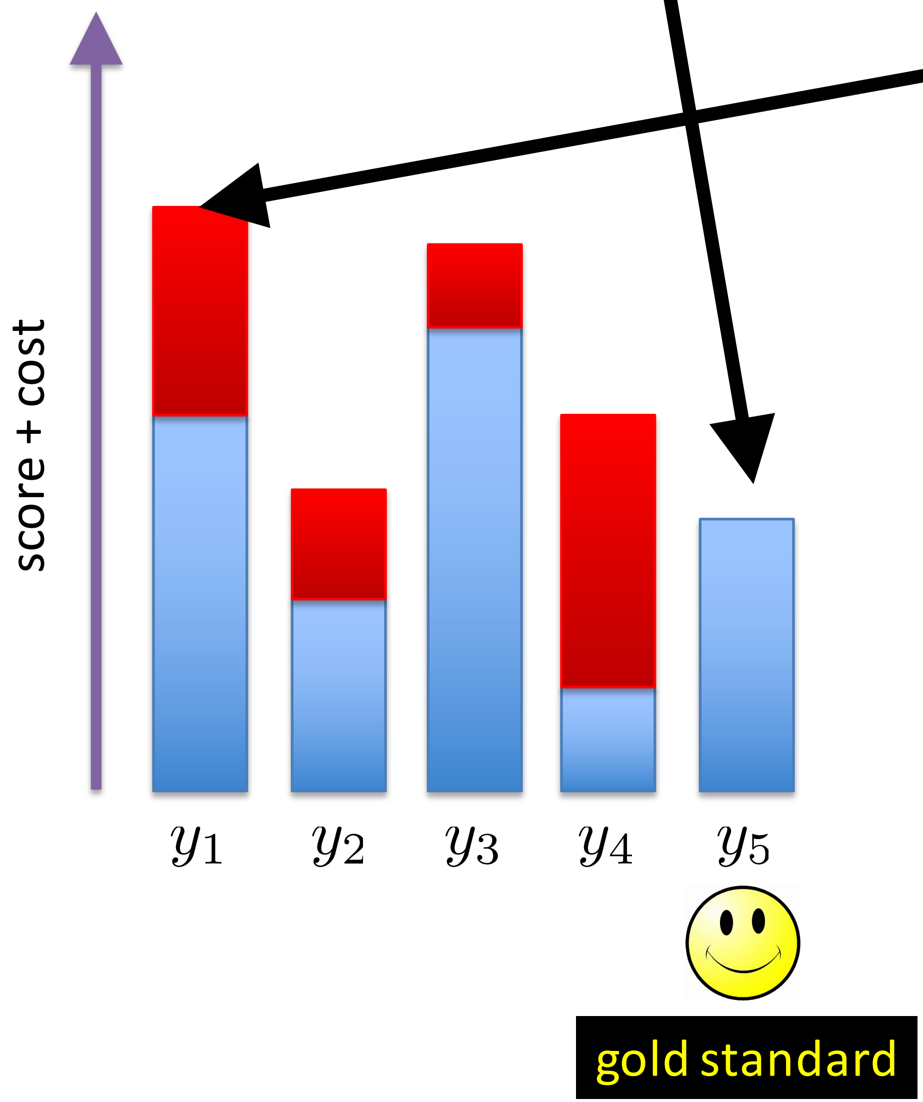
hinge loss:

$$\mathrm{loss}_{\mathrm{hinge}}(\boldsymbol{x}, y, \boldsymbol{\theta}) = -\mathrm{score}(\boldsymbol{x}, y, \boldsymbol{\theta}) + \underbrace{\max_{y' \in \mathcal{L}} \left(\mathrm{score}(\boldsymbol{x}, y', \boldsymbol{\theta}) + \mathrm{cost}(y, y')\right)}$$



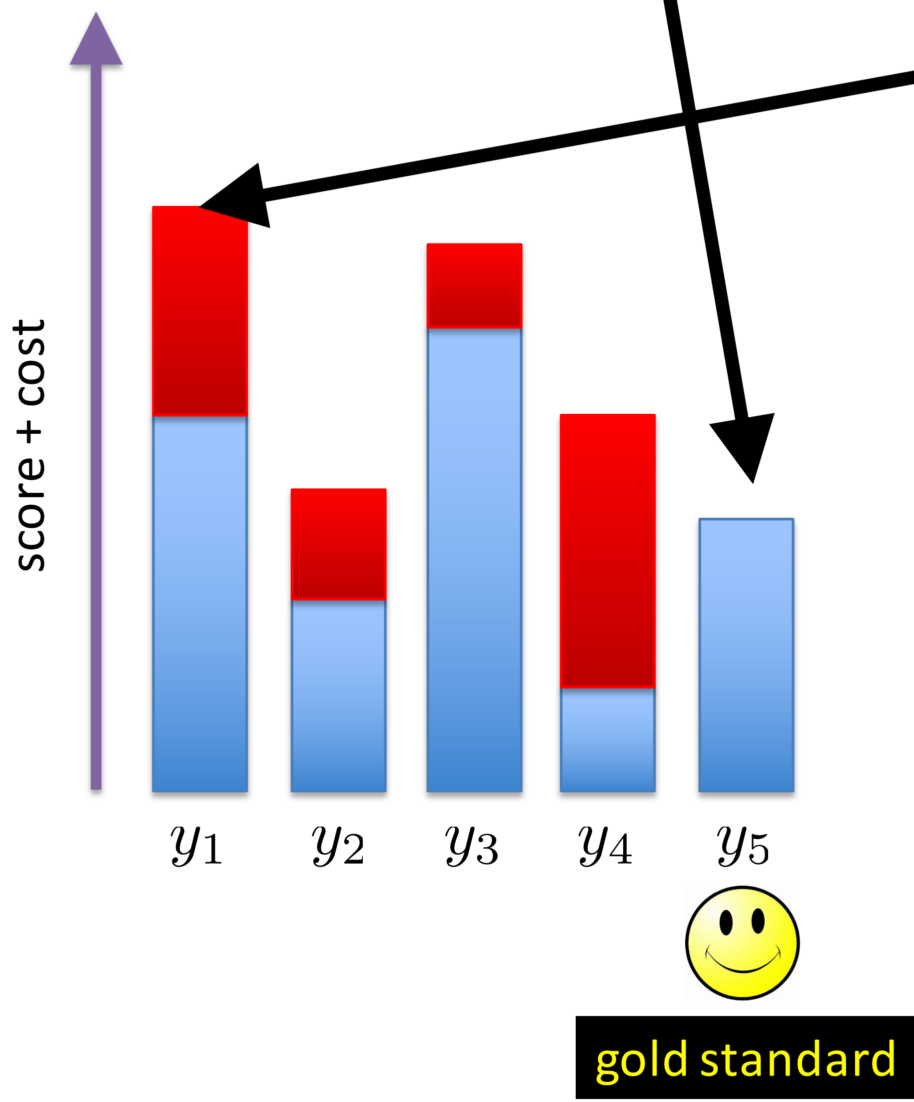
hinge loss:

$$\text{loss}_{\text{hinge}}(\boldsymbol{x}, y, \boldsymbol{\theta}) = -\text{score}(\boldsymbol{x}, y, \boldsymbol{\theta}) + \underbrace{\max_{y' \in \mathcal{L}} \left(\text{score}(\boldsymbol{x}, y', \boldsymbol{\theta}) + \text{cost}(y, y')\right)}$$



hinge loss:

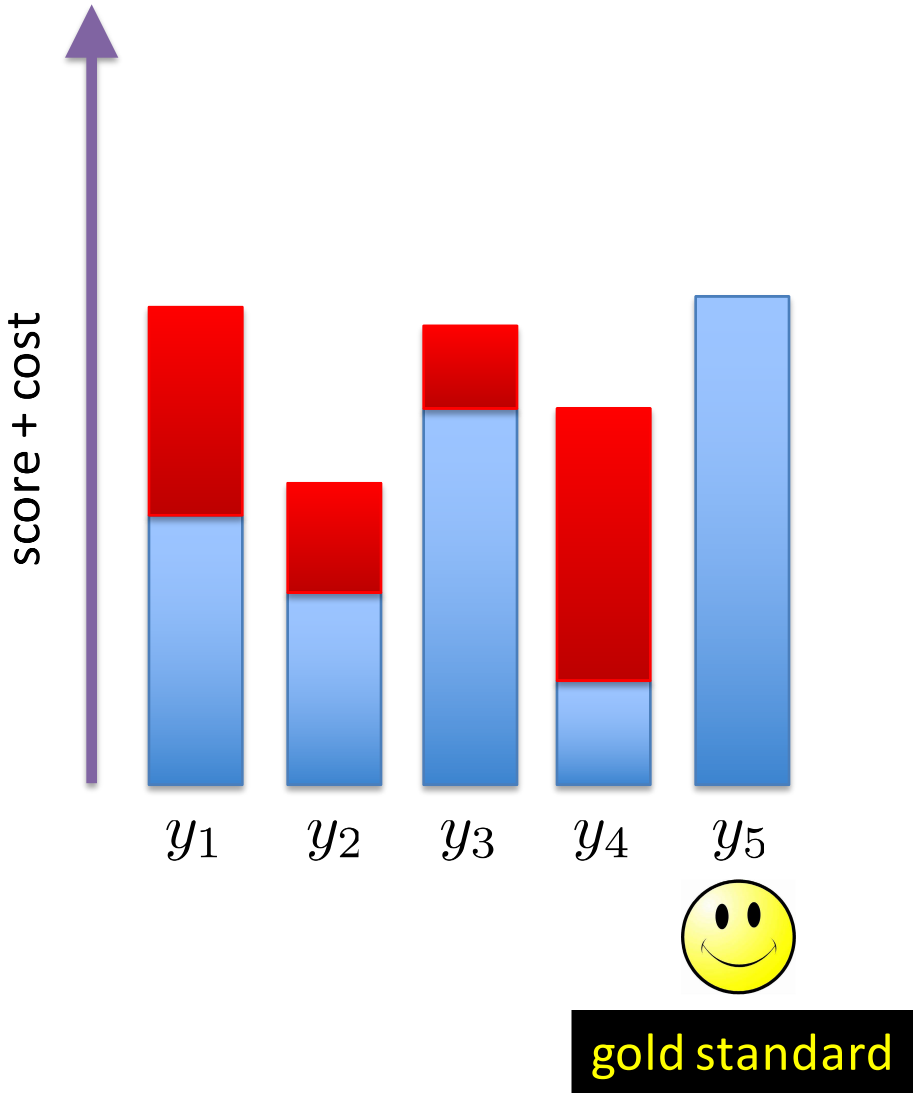
$$\text{loss}_{\text{hinge}}(\boldsymbol{x}, y, \boldsymbol{\theta}) = -\text{score}(\boldsymbol{x}, y, \boldsymbol{\theta}) + \underbrace{\max_{y' \in \mathcal{L}} \left(\text{score}(\boldsymbol{x}, y', \boldsymbol{\theta}) + \text{cost}(y, y')\right)}$$



effect of learning?

hinge loss:

$$\text{loss}_{\text{hinge}}(\boldsymbol{x}, y, \boldsymbol{\theta}) = -\text{score}(\boldsymbol{x}, y, \boldsymbol{\theta}) + \max_{y' \in \mathcal{L}} \left(\text{score}(\boldsymbol{x}, y', \boldsymbol{\theta}) + \text{cost}(y, y')\right)$$



effect of learning:
score of gold standard will be higher than score+cost of all others

Subgradients of Loss Functions

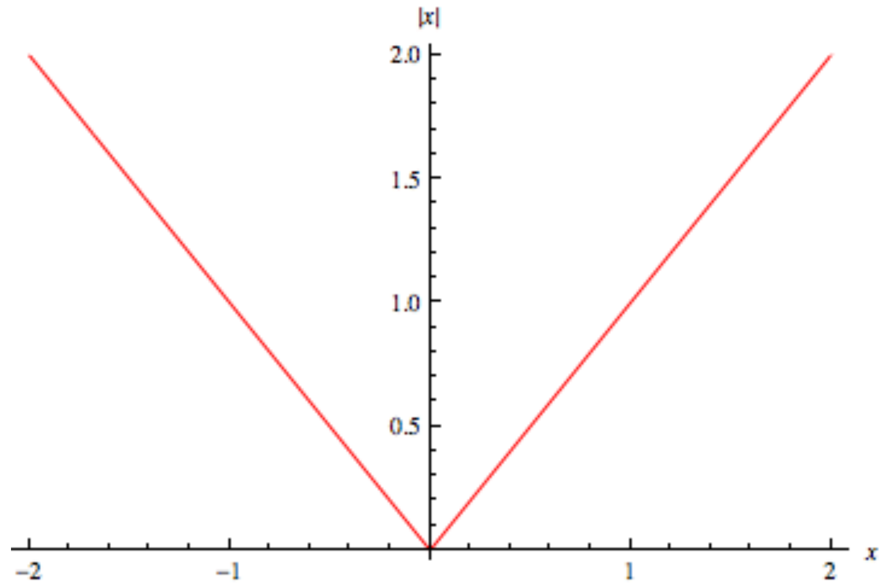
- some of our loss functions are not differentiable:

$$\text{loss}_{\text{perc}}(\boldsymbol{x}, y, \boldsymbol{\theta}) = -\sum_i \theta_i f_i(\boldsymbol{x}, y) + \max_{y' \in \mathcal{L}} \sum_i \theta_i f_i(\boldsymbol{x}, y')$$

- but they are subdifferentiable:

Subgradient Examples

$$f(x) = |x| = \max(x, -x)$$



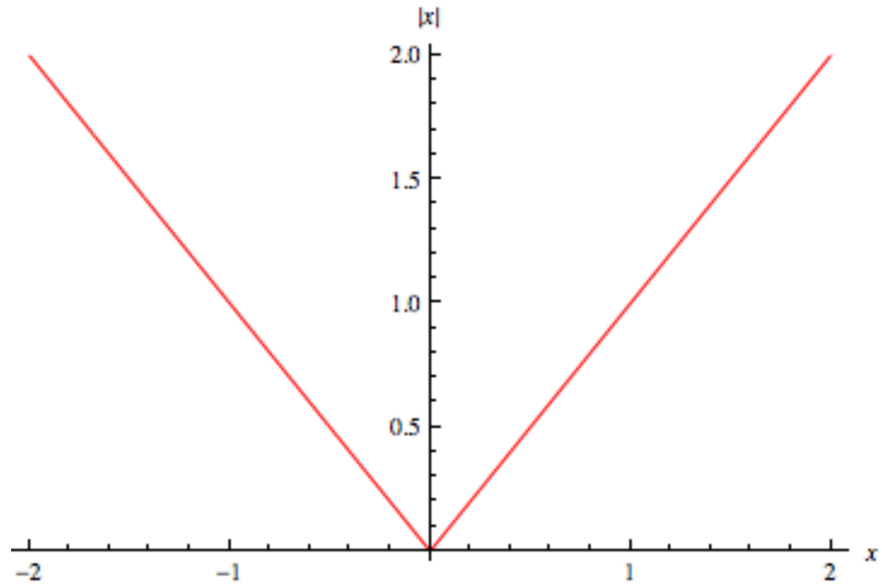
$$x < 0: \ \ \partial f(x) =$$

$$x > 0: \ \ \partial f(x) =$$

$$x = 0: \ \ \partial f(x) =$$

Subgradient Examples

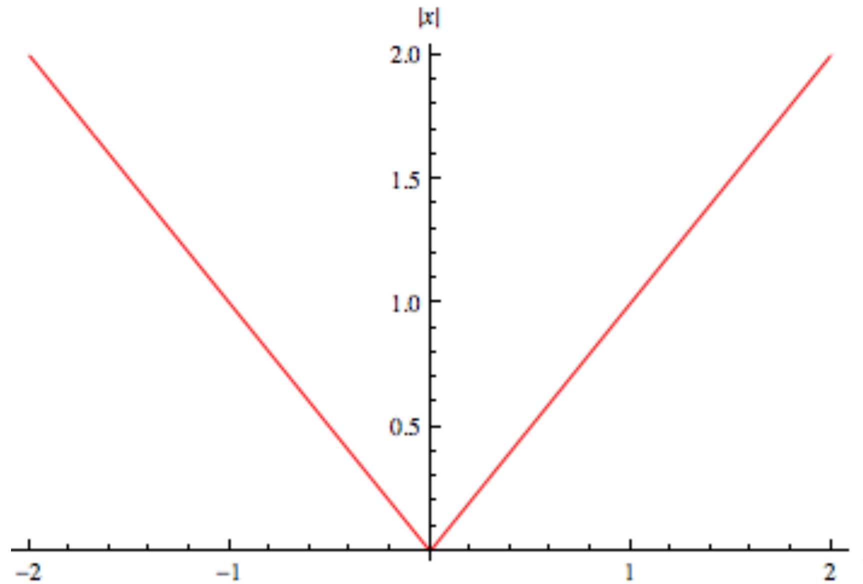
$$f(x) = |x| = \max(x, -x)$$



$$x < 0{:}\quad \partial f(x) = \{-1\}$$
$$x > 0{:}\quad \partial f(x) = \{1\}$$
$$x = 0{:}\quad \partial f(x) =$$

Subgradient Examples

$$f(x) = |x| = \max(x, -x)$$



$$x < 0: \quad \partial f(x) = \{-1\}$$
$$x > 0: \quad \partial f(x) = \{1\}$$
$$x = 0: \quad \partial f(x) = [-1, 1]$$

- to find a subgradient of max of convex functions at a point, choose one function that achieves the max at that point and choose any of its subgradients at the point

Subgradients of Loss Functions

- some of our loss functions are not differentiable:

$$\mathrm{loss}_{\mathrm{perc}}(\boldsymbol{x}, y, \boldsymbol{\theta}) = -\sum_i \theta_i f_i(\boldsymbol{x}, y) + \max_{y' \in \mathcal{L}} \sum_i \theta_i f_i(\boldsymbol{x}, y')$$

- but they are subdifferentiable:

Subgradients of Loss Functions

- some of our loss functions are not differentiable:

$$\text{loss}_{\text{perc}}(\boldsymbol{x}, y, \boldsymbol{\theta}) = -\sum_i \theta_i f_i(\boldsymbol{x}, y) + \max_{y' \in \mathcal{L}} \sum_i \theta_i f_i(\boldsymbol{x}, y')$$

- but they are subdifferentiable:

$$\frac{\partial \text{loss}_{\text{perc}}(\boldsymbol{x}, y, \boldsymbol{\theta})}{\partial \theta_j} = -f_j(\boldsymbol{x}, y) + f_j(\boldsymbol{x}, \text{classify}(\boldsymbol{x}, \boldsymbol{\theta}))$$

Subgradients of Loss Functions

$$\text{loss}_{\text{perc}}(\boldsymbol{x}, y, \boldsymbol{\theta}) = -\sum_i \theta_i f_i(\boldsymbol{x}, y) + \max_{y' \in \mathcal{L}} \sum_i \theta_i f_i(\boldsymbol{x}, y')$$

$$\frac{\partial \text{loss}_{\text{perc}}(\boldsymbol{x}, y, \boldsymbol{\theta})}{\partial \theta_j} = -f_j(\boldsymbol{x}, y) + f_j(\boldsymbol{x}, \text{classify}(\boldsymbol{x}, \boldsymbol{\theta}))$$

$$\frac{\partial}{\partial \theta_j} \max_{y' \in \mathcal{L}} \sum_i \theta_i f_i(\boldsymbol{x}, y') =$$

find subgradient of the
function that achieves the max

Subgradients of Loss Functions

$$\text{loss}_{\text{perc}}(\boldsymbol{x}, y, \boldsymbol{\theta}) = -\sum_i \theta_i f_i(\boldsymbol{x}, y) + \max_{y' \in \mathcal{L}} \sum_i \theta_i f_i(\boldsymbol{x}, y')$$

$$\frac{\partial \text{loss}_{\text{perc}}(\boldsymbol{x}, y, \boldsymbol{\theta})}{\partial \theta_j} = -f_j(\boldsymbol{x}, y) + f_j(\boldsymbol{x}, \text{classify}(\boldsymbol{x}, \boldsymbol{\theta}))$$

$$\frac{\partial}{\partial \theta_j} \max_{y' \in \mathcal{L}} \sum_i \theta_i f_i(\boldsymbol{x}, y') = \frac{\partial}{\partial \theta_j} \sum_i \theta_i f_i(\boldsymbol{x}, \text{classify}(\boldsymbol{x}, \boldsymbol{\theta})) = f_j(\boldsymbol{x}, \text{classify}(\boldsymbol{x}, \boldsymbol{\theta}))$$

find subgradient of the
function that achieves the max

Roadmap

- classification
- words
- lexical semantics
- language modeling
- sequence labeling
- syntax and syntactic parsing
- neural network methods in NLP
- semantic compositionality
- semantic parsing
- unsupervised learning
- machine translation and other applications

Words

- what is a word?
- tokenization
- morphology
- word sense

What is a word?

Tokenization

- tokenization: convert a character stream into words by adding spaces
- for certain languages, highly nontrivial
- e.g., Chinese word segmentation is a widely-studied NLP task

Tokenization

- for other languages (English), tokenization is easier but is still not always obvious
- the data for your homework has been tokenized:
  - punctuation has been split off from words
  - contractions have been split

Intricacies of Tokenization

- separating punctuation characters from words?
  - *, ” ? !* → always separate
  - *.* → when shouldn’t we separate it?

Intricacies of Tokenization

- separating punctuation characters from words?
  - *, ” ? !* → always separate
  - *.* → when shouldn’t we separate it?
    - *Dr., Mr., Prof., U.S., etc.*

Intricacies of Tokenization

- separating punctuation characters from words?
  - *, ” ? !* → always separate
  - *.* → when shouldn’t we separate it?
    - *Dr., Mr., Prof., U.S., etc.*
- English contractions:
  - *isn’t, aren’t, wasn’t,…* → *is n’t, are n’t, was n’t,…*
  - but how about these: *can’t, won’t* → *ca n’t, wo n’t*
  - *ca* and *wo* are then different forms from *can* and *will*

- Chinese and Japanese: no spaces between words:
  - 莎拉波娃现在居住在美国东南部的佛罗里达。
  - 莎拉波娃 现在 居住 在 美国 东南部 的 佛罗里达
  - Sharapova now lives in US southeastern Florida
- Further complicated in Japanese, with multiple alphabets intermingled
  - Dates/amounts in multiple formats

フォーチュン500社は情報不足のため時間あた$500K(約6,000万円)

Katakana Hiragana Kanji Romaji

End-user can express query entirely in hiragana!

J&M/SLP3

Word Segmentation in Chinese

- Chinese words are composed of characters
  - characters are generally 1 syllable and 1 morpheme
  - average word is 2.4 characters long
- standard baseline segmentation algorithm:
  - Maximum Matching (also called Greedy)

Maximum Matching
Word Segmentation Algorithm

Given a Chinese word list and a string:

1) start a pointer at the beginning of the string
2) find longest word in dictionary that matches the string starting at pointer
3) move the pointer over the word in string
4) go to 2

Maximum Matching Examples

- Thecatinthehat — the cat in the hat
- Thetabledownthere — the table down there
  theta bled own there

- Doesn’t generally work in English!

- But works astonishingly well in Chinese
  - 莎拉波娃现在居住在美国东南部的佛罗里达。
  - 莎拉波娃 现在 居住 在 美国 东南部 的 佛罗里达
- Modern probabilistic segmentation algorithms even better

Effect on Machine Translation

Reference translation:
scientists complete sequencing of the chromosome linked to early dementia

CharBased segmented input:
科␣学␣家␣为␣攸␣关␣初␣期␣失␣智␣症␣的␣染␣色␣体␣完␣成␣定␣序

MaxMatch segmented input:
科学家␣为␣攸关␣初期␣失␣智␣症␣的␣染色␣体␣完成␣定␣序

Translation with CharBased segmentation:
scientists at the beginning of the stake of chile lost the genome sequence completed

Translation with MaxMatch segmentation:
scientists at stake for the early loss of intellectual syndrome chromosome completed sequencing

Chang, Galley, & Manning (2008)

Removing Spaces?

- tokenization is usually about adding spaces
- but might we also want to remove spaces?
- what are some English examples?

Removing Spaces?

- tokenization is usually about adding spaces
- but might we also want to remove spaces?
- what are some English examples?
  - names?
    - New York → NewYork
  - non-compositional compounds?
    - hot dog → hotdog
  - other artifacts of our spacing conventions?
    - New York-Long Island Railway

Types and Tokens

- once text has been tokenized, let's count the words
- types: entries in the vocabulary
- tokens: instances of types in a corpus
- example sentence: *If they want to go , they should go .*
  - how many types?
  - how many tokens?

Types and Tokens

- once text has been tokenized, let's count the words
- types: entries in the vocabulary
- tokens: instances of types in a corpus
- example sentence: *If they want to go , they should go .*
  - how many types? 8
  - how many tokens? 10
- type/token ratio: useful statistic of a corpus (here, 0.8)

Types and Tokens

- once text has been tokenized, let’s count the words
- types: entries in the vocabulary
- tokens: instances of types in a corpus
- example sentence: *If they want to go , they should go .*
  - how many types? 8
  - how many tokens? 10
- type/token ratio: useful statistic of a corpus (here, 0.8)
- as we add data, what happens to the type-token ratio?
- indicates what?
  - high type/token ratio →
  - low type/token ratio →

Types and Tokens

- once text has been tokenized, let's count the words
- types: entries in the vocabulary
- tokens: instances of types in a corpus
- example sentence: *If they want to go , they should go .*
  - how many types? 8
  - how many tokens? 10
- type/token ratio: useful statistic of a corpus (here, 0.8)
- as we add data, what happens to the type-token ratio?
- indicates what?
  - high type/token ratio → rich morphology
  - low type/token ratio → poor morphology

How many words are there?

- how many English words exist?
- when we increase the size of our corpus, what happens to the number of types?

How many words are there?

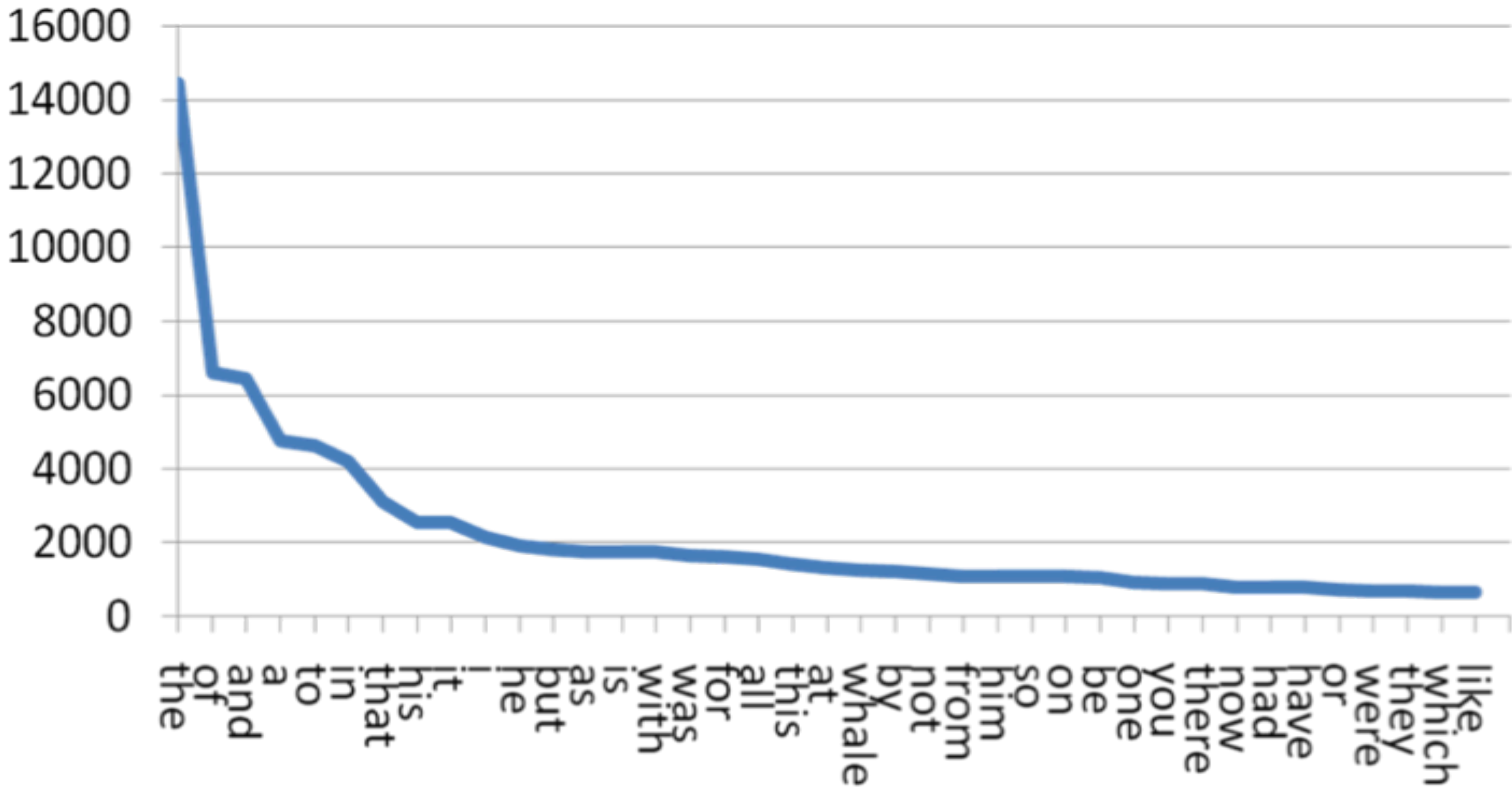
- how many English words exist?
- when we increase the size of our corpus, what happens to the number of types?
  – a bit surprising: vocabulary continues to grow in any actual dataset
  – you'll just never see all the words
  – in 1 million tweets, 15M tokens, 600k types
  – in 56 million tweets, 847M tokens, ? types

How many words are there?

- how many English words exist?
- when we increase the size of our corpus, what happens to the number of types?
  - a bit surprising: vocabulary continues to grow in any actual dataset
  - you’ll just never see all the words
  - in 1 million tweets, 15M tokens, 600k types
  - in 56 million tweets, 847M tokens, 11M types

How are words distributed?

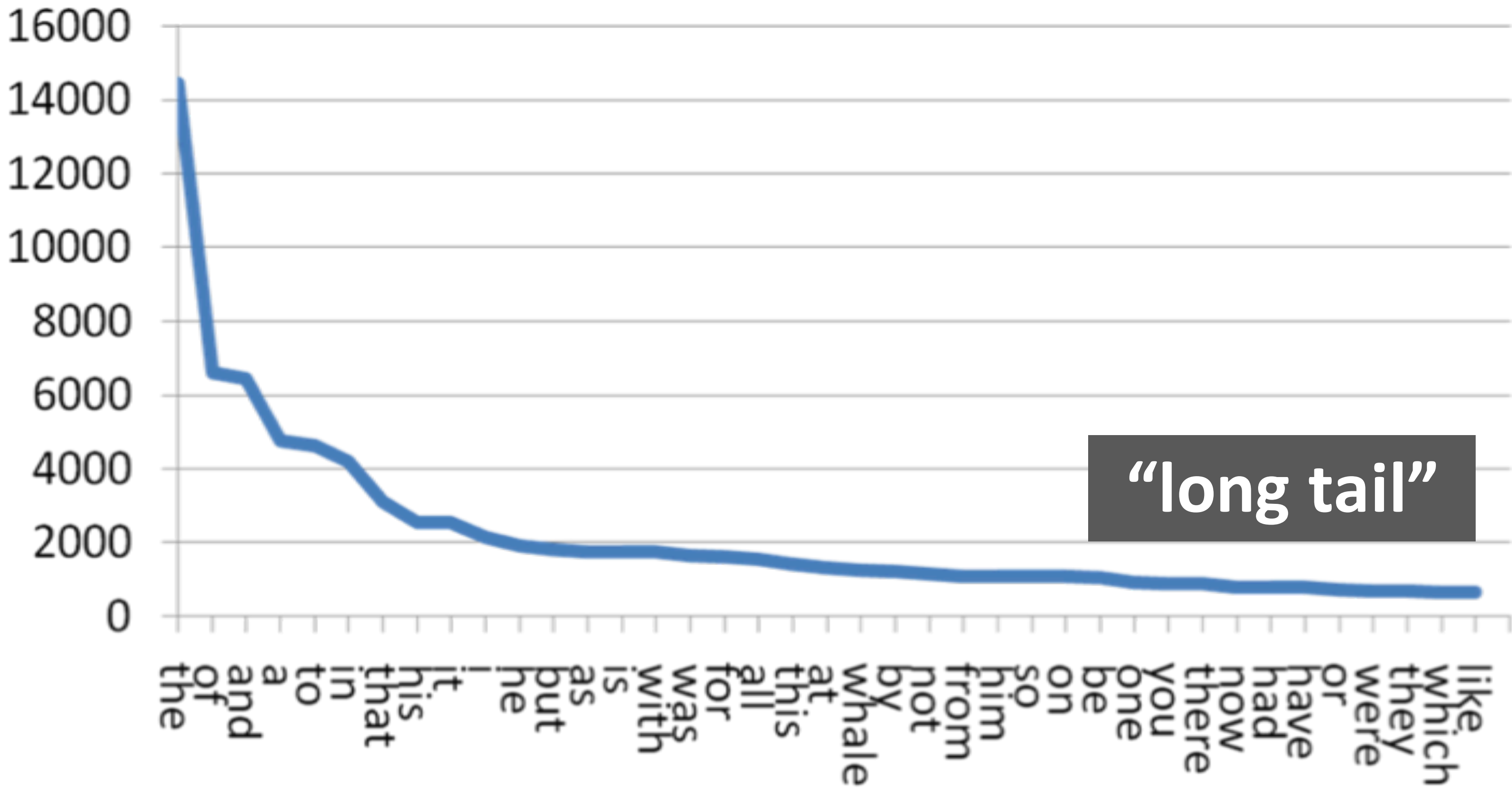
- Zipf's law: frequency of a word is inversely proportional to its rank



blog EMIS

How are words distributed?

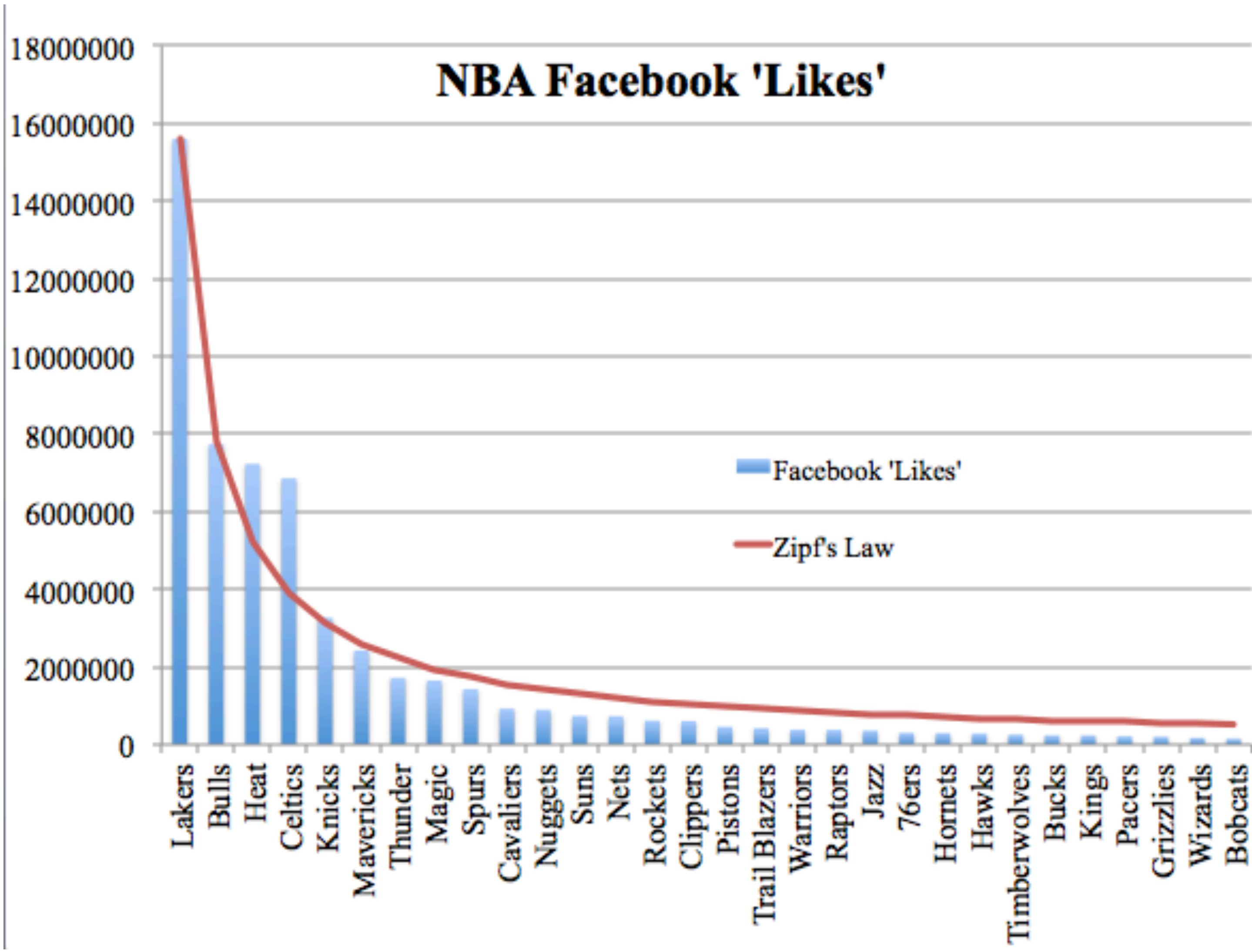
- Zipf's law: frequency of a word is inversely proportional to its rank



blog EMIS

Zipf's Law

- also predicts other kinds of data: population of cities in a country, revenue of different companies, etc.



The Laurentian University Sports Analytics Group

The Long Tail

- there are so many word types!
- but words have **internal structure**

Morphology

- morphemes:
  - the small meaningful units that make up words
  - stems: core meaning-bearing units
  - affixes: bits and pieces that adhere to stems
    - often with grammatical functions

Kinds of Word Formation

- inflection: modifying a word with an affix to change its grammatical function (tense, number, etc.)
  - result is a "different form of the same word"
  - examples: *book* → *books*, *walk* → *walked*

Kinds of Word Formation

- inflection: modifying a word with an affix to change its grammatical function (tense, number, etc.)
  - result is a "different form of the same word"
  - examples: *book* → *books*, *walk* → *walked*
- derivation: adding an affix to a stem to create a new word
  - examples: *great* → *greatly*, *great* → *greatness*

Kinds of Word Formation

- inflection: modifying a word with an affix to change its grammatical function (tense, number, etc.)
  - result is a "different form of the same word"
  - examples: *book* → *books*, *walk* → *walked*
- derivation: adding an affix to a stem to create a new word
  - examples: *great* → *greatly*, *great* → *greatness*
- compounding: combining two stems
  - examples: *lawsuit, keyboard, bookcase*

Morphology

- usually, morphological derivation is simply splitting a word into its morphemes:
  - walked = walk + ed
  - greatness = great + ness
- but it can actually be a hierarchical structure

Morphology

- ambiguity in hierarchical morphological decomposition?
  - rare, but it does happen

Morphology

- ambiguity in hierarchical morphological decomposition?
  - rare, but it does happen
  - unlockable = un + lock + able
    - what does this word mean?

Morphology

- ambiguity in hierarchical morphological decomposition?
  - rare, but it does happen
  - unlockable = un + lock + able
    - what does this word mean?
    - (un+lock)+able: “able to be unlocked”
    - un+(lock+able): “unable to be locked”

Morphology in NLP

- two common tasks:
  - lemmatization
  - stemming

Lemmatization

- lemmatization: reduce inflections or variant forms to base form
  - *am, are, is* → *be*
  - *car, cars, car's, cars'* → *car*
- *the boy's cars are different colors* → *the boy car be different color*
- have to find correct dictionary headword form
- e.g., for machine translation:
  - Spanish *quiero* ('I want'), *quieres* ('you want') same lemma as *querer* 'want'

J&M/SLP3

Stemming

- stemming: reduces words to their stems via crude chopping of affixes
  - e.g., *automate(s), automatic, automation* all reduced to *automat*
  - language dependent
  - key step in information retrieval

for example compressed and compression are both accepted as equivalent to compress.

for exampl compress and compress ar both accept as equival to compress

J&M/SLP3

Porter's algorithm
The most common English stemmer

Step 1a

```
sses → ss   caresses → caress
ies  → i    ponies   → poni
ss   → ss   caress   → caress
s    → ø    cats     → cat
```

Step 1b

```
(*v*)ing → ø  walking   → walk
              sing      → sing
(*v*)ed  → ø  plastered → plaster
…
```

Step 2 (for long stems)

```
ational→ ate  relational→ relate
izer→ ize     digitizer → digitize
ator→ ate     operator  → operate
…
```

Step 3 (for longer stems)

```
al    → ø  revival    → reviv
able  → ø  adjustable → adjust
ate   → ø  activate   → activ
…
```

J&M/SLP3

Viewing morphology in a corpus
Why only strip –ing if there is a vowel?

```
(*v*)ing → ø  walking   → walk
              sing      → sing
```

Viewing morphology in a corpus
Why only strip –ing if there is a vowel?

```
(*v*)ing → ø  walking  → walk
              sing     → sing
```

```
tr -sc 'A-Za-z' '\n' < shakes.txt | grep 'ing$' | sort | uniq -c | sort –nr
```

1312	King	548	being
548	being	541	nothing
541	nothing	152	something
388	king	145	coming
375	bring	130	morning
358	thing	122	having
307	ring	120	living
152	something	117	loving
145	coming	116	Being
130	morning	102	going

```
tr -sc 'A-Za-z' '\n' < shakes.txt | grep '[aeiou].*ing$' | sort | uniq -c | sort –nr
```

J&M/SLP3

Dealing with complex morphology is sometimes necessary

- Some languages requires complex morpheme segmentation
  - Turkish
  - *Uygarlastiramadiklarimizdanmissinizcasina*: "(behaving) as if you are among those whom we could not civilize"
  - Uygar `civilized' + las `become'

    + tir `cause' + ama `not able'

    + dik `past' + lar 'plural'

    + imiz 'p1pl' + dan 'abl'

    + mis 'past' + siniz '2pl' + casina 'as if'

J&M/SLP3

Terminology: lemma and wordform

- lemma or citation form
  - same stem, part of speech, rough semantics
- wordform
  - inflected word as it appears in text

wordform	lemma
banks	bank
sung	sing
duermes	dormir