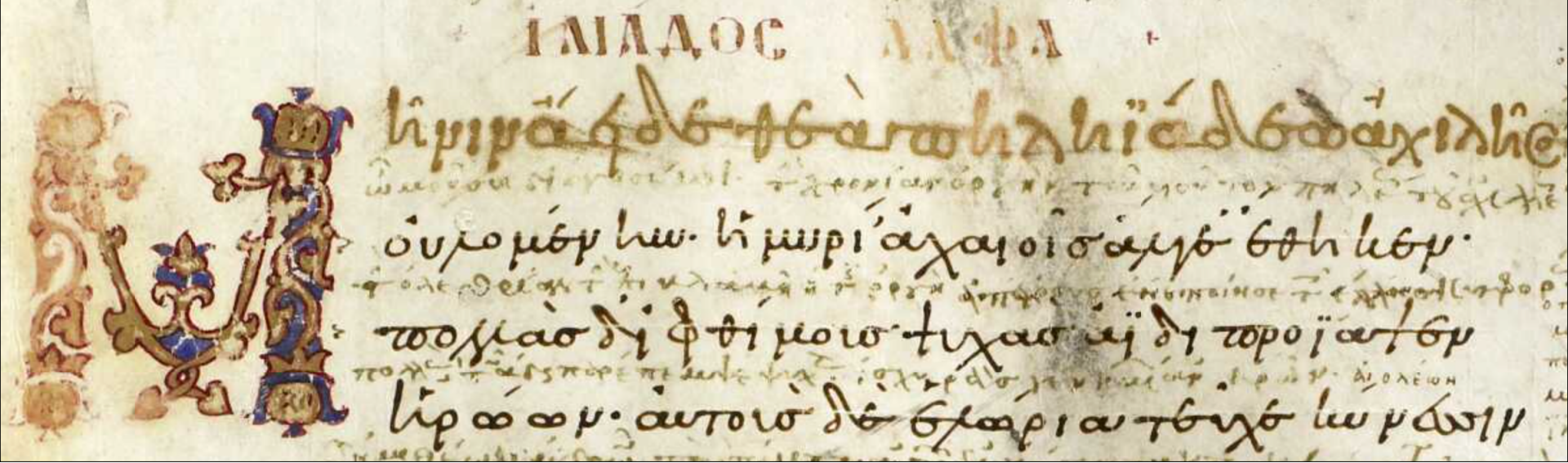




Natural Language Processing

Info 159/259

Lecture 3: Text classification 1 (Jan 25, 2022)

David Bamman, UC Berkeley



Classification

A mapping h from input data x (drawn from instance space $\mathcal{X}$) to a label (or labels) y from some enumerable output space $\mathcal{Y}$

$\mathcal{X}$ = set of all documents

$\mathcal{Y} = \{\text{english, mandarin, greek, ...}\}$

x = a single document

y = ancient greek



Classification

$$h(x) = y$$

$h(\mu\eta\eta\nu\nu\ \acute{\alpha}\epsilon\iota\delta\epsilon\ \theta\epsilon\acute{\alpha}) = \text{ancient grc}$



Classification

Let $h(x)$ be the “true” mapping.
We never know it. How do we
find the best $\hat{h}(x)$ to
approximate it?

One option: rule based

if x has characters in
unicode point range 0370-03FF:
 $\hat{h}(x) = \text{greek}$



Classification

Supervised learning

Given training data in the form
of $\langle x, y \rangle$ pairs, learn $\hat{h}(x)$

Text categorization problems

task	$\mathcal{X}$	$\mathcal{Y}$
language ID	text	{english, mandarin, greek, ...}
spam classification	email	{spam, not spam}
authorship attribution	text	{jk rowling, james joyce, ...}
genre classification	novel	{detective, romance, gothic, ...}
sentiment analysis	text	{positive, negative, neutral, mixed}

Sentiment analysis

- Document-level SA: is the entire text **positive** or **negative** (or both/neither) with respect to an implicit target?
- Movie reviews [Pang et al. 2002, Turney 2002]

Training data

positive

“... is a film which still causes real, not figurative, chills to run along my spine, and it is certainly the bravest and most ambitious fruit of Coppola's genius”

Roger Ebert, Apocalypse Now

- “I hated this movie. Hated hated hated hated hated this movie. Hated it. Hated every simpering stupid vacant audience-insulting moment of it. Hated the sensibility that thought anyone would like it.”

negative

Roger Ebert, North

Sentiment analysis

- Is the text positive or negative (or both/neither) with respect to an explicit target **within the text**?

Feature: **picture**

Positive: 12

- Overall this is a good camera with a really good picture clarity.
- The pictures are absolutely amazing - the camera captures the minutest of details.
- After nearly 800 pictures I have found that this camera takes incredible pictures.

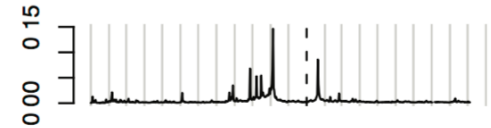
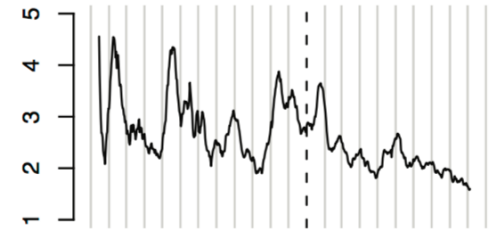
...

Negative: 2

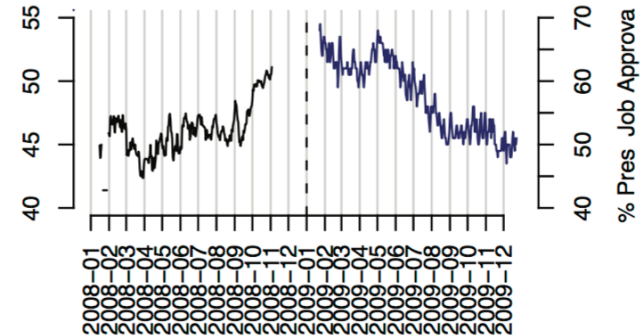
- The pictures come out hazy if your hands shake even for a moment during the entire process of taking a picture.
- Focusing on a display rack about 20 feet away in a brightly lit room during day time, pictures produced by this camera were blurry and in a shade of orange.

Hu and Liu (2004), "Mining and Summarizing Customer Reviews"

Twitter sentiment →



Job approval polls →



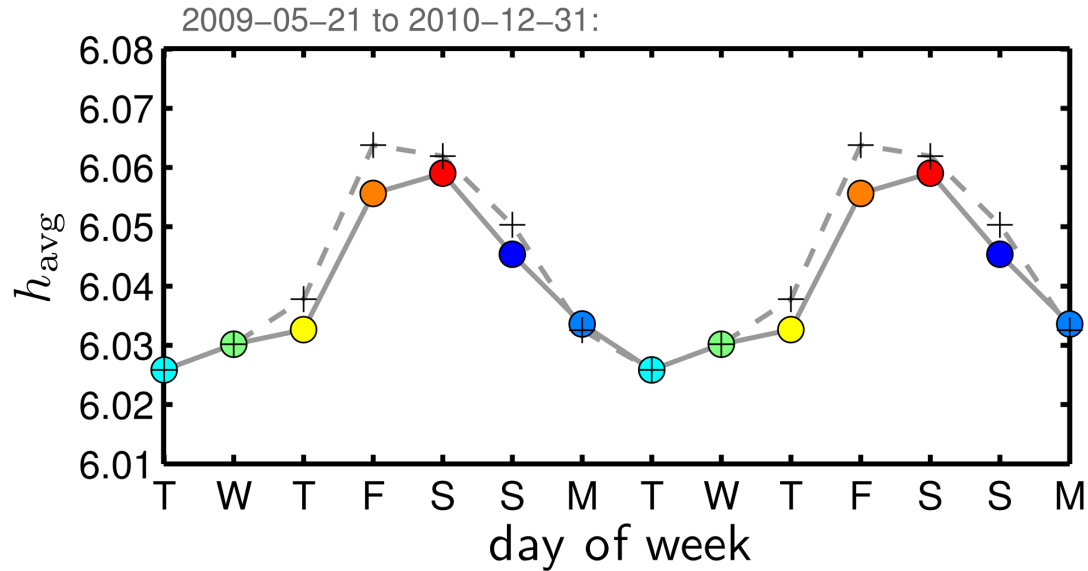
O'Connor et al (2010), "From Tweets to Polls: Linking Text Sentiment to Public Opinion Time Series"

Figure 9: The sentiment ratio for *obama* (15-day window), and fraction of all Twitter messages containing *obama* (day-by-day, no smoothing), compared to election polls (2008) and job approval polls (2009).

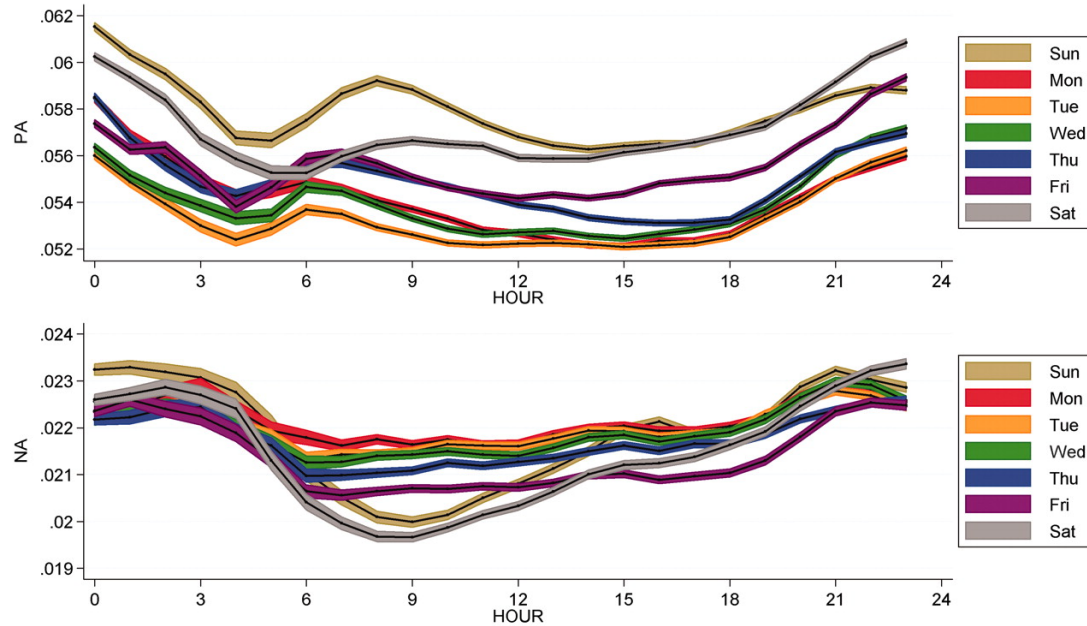
Sentiment as tone

- No longer the speaker's attitude with respect to some particular target, but rather the positive/negative **tone** that is evinced.

Sentiment as tone



Sentiment as tone



Golder and Macy (2011), "Diurnal and Seasonal Mood Vary with Work, Sleep, and Daylength Across Diverse Cultures," *Science*. Positive affect (PA) and negative affect (NA) measured with LIWC.

Sentiment Dictionaries

- General Inquirer (1966)
- MPQA subjectivity lexicon (Wilson et al. 2005)
http://mpqa.cs.pitt.edu/lexicons/subj_lexicon/
- LIWC (Linguistic Inquiry and Word Count, Pennebaker 2015)
- AFINN (Nielsen 2011)
- NRC Word-Emotion Association Lexicon (EmoLex), Mohammad and Turney 2013

pos	neg
unlimited	lag
prudent	contortions
superb	fright
closeness	lonely
impeccably	tenuously
fast-paced	plebeian
treat	mortification
destined	outrage
blessing	allegations
steadfastly	disoriented

LIWC

- 73 separate lexicons designed for applications social psychology

Positive Emotion	Negative Emotion	Insight	Inhibition	Family	Negate
appreciat*	anger*	aware*	avoid*	brother*	aren't
comfort*	bore*	believe	careful*	cousin*	cannot
great	cry	decid*	hesitat*	daughter*	didn't
happy	despair*	feel	limit*	family	neither
interest	fail*	figur*	oppos*	father*	never
joy*	fear	know	prevent*	grandf*	no
perfect*	griev*	knew	reluctan*	grandm*	nobod*
please*	hate*	means	safe*	husband	none
safe*	panic*	notice*	stop	mom	nor
terrific	suffers	recogni*	stubborn*	mother	nothing
value	terrify	sense	wait	niece*	nowhere
wow*	violent*	think	wary	wife	without

Why is SA hard?

- Sentiment is a measure of a speaker's private state, which is unobservable.
- Sometimes words are a good indicator of sentiment (love, amazing, hate, terrible); many times it requires deep world + contextual knowledge

“*Valentine's Day* is being marketed as a Date Movie. I think it's more of a First-Date Movie. If your date likes it, do not date that person again. And if you like it, there may not be a second date.”

Roger Ebert, *Valentine's Day*



Classification

Supervised learning

Given training data in the form of $\langle x, y \rangle$ pairs, learn $\hat{h}(x)$

x	y
loved it!	positive
terrible movie	negative
not too shabby	positive

$$\hat{h}(x)$$

- The classification function that we want to learn has two different components:
 - the formal structure of the learning method (what's the relationship between the input and output?) → Naive Bayes, logistic regression, convolutional neural network, etc.
 - the **representation** of the data

Representation for SA

- Only positive/negative words in MPQA
- Only words in isolation (bag of words)
- Conjunctions of words (sequential, skip ngrams, other non-linear combinations)
- Higher-order linguistic structure (e.g., syntax)

“... is a film which still causes real, not figurative, chills to run along my spine, and it is certainly the **bravest** and most **ambitious** fruit of Coppola's **genius**”

Roger Ebert, Apocalypse Now

“I **hated** this movie. **Hated hated hated hated hated** this movie. **Hated** it. **Hated** every simpering **stupid** vacant audience-**insulting** moment of it. **Hated** the sensibility that thought anyone would **like** it.”

Roger Ebert, North

Bag of words

Representation of text only
as the counts of words that
it contains (or a binary
indicator of the presence/
absence of that word).

	Apocalypse now	North
the	1	1
of	0	0
hate	0	9
genius	1	0
bravest	1	0
stupid	0	1
like	0	1
...		

Remember

$$\sum_{i=1}^F x_i \beta_i = x_1 \beta_1 + x_2 \beta_2 + \dots + x_F \beta_F$$

$$\prod_{i=1}^F x_i = x_1 \times x_2 \times \dots \times x_F$$

$$\exp(x) = e^x \approx 2.7^x$$

$$\exp(x + y) = \exp(x) \exp(y)$$

$$\log(x) = y \rightarrow e^y = x$$

$$\log(xy) = \log(x) + \log(y)$$

Binary logistic regression

$$P(y = 1 \mid x, \beta) = \frac{1}{1 + \exp \left(- \sum_{i=1}^F x_i \beta_i \right)}$$

output space

$$\mathcal{Y} = \{0, 1\}$$

x = feature vector

Feature	Value
the	0
and	0
bravest	0
love	0
loved	0
genius	0
not	0
fruit	1
<i>BIAS</i>	1

β = coefficients

Feature	β
the	0.01
and	0.03
bravest	1.4
love	3.1
loved	1.2
genius	0.5
not	-3.0
fruit	-0.8
<i>BIAS</i>	-0.1

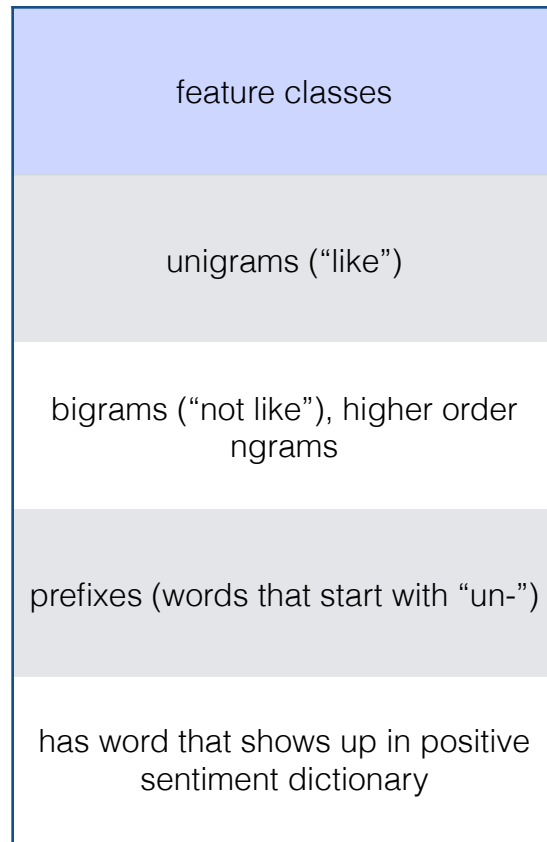
Features

- As a discriminative classifier, logistic regression doesn't assume features are independent.
- Its power partly comes in the ability to create richly expressive features without the burden of independence.
- We can represent text through features that are not just the identities of individual words, but any feature that is scoped over **the entirety of the input**.

features
contains like
has word that shows up in positive sentiment dictionary
review begins with "I like"
at least 5 mentions of positive affectual verbs (like, love, etc.)

Features

- Features are where you can encode your own **domain understanding** of the problem.



Features

Feature	Value
the	0
and	0
bravest	0
love	0
loved	0
genius	0
not	1
fruit	0
<i>BIAS</i>	1

Feature	Value
like	1
not like	1
did not like	1
in_pos_dict_MPQA	1
in_neg_dict_MPQA	0
in_pos_dict_LIWC	1
in_neg_dict_LIWC	0
author=ebert	1
author=siskel	0

β = coefficients

How do we get good
values for β ?

Feature	β
the	0.01
and	0.03
bravest	1.4
love	3.1
loved	1.2
genius	0.5
not	-3.0
fruit	-0.8
<i>BIAS</i>	-0.1

Conditional likelihood

$$\prod_i^N P(y_i | x_i, \beta)$$

For all training data, we want the probability of the **true label y** for each data point **x** to be high

	BIAS	love	loved	$a = \sum x_i \beta_i$	$\exp(-a)$	$1/(1 + \exp(-a))$	true y
x^1	1	1	0	3	0.05	95.2%	1
x^2	1	1	1	4.2	0.015	98.5%	1
x^3	1	0	0	-0.1	1.11	47.5%	0

Conditional likelihood

$$\prod_i^N P(y_i | x_i, \beta)$$

For all training data, we want the probability of the true label y for each data point x to be high

This principle gives us a way to pick the values of the parameters β that maximize the probability of the training data $\langle x, y \rangle$

The value of β that maximizes likelihood also maximizes the log likelihood

$$\arg \max_{\beta} \prod_{i=1}^N P(y_i | x_i, \beta) = \arg \max_{\beta} \log \prod_{i=1}^N P(y_i | x_i, \beta)$$

The log likelihood is an easier form to work with:

$$\log \prod_{i=1}^N P(y_i | x_i, \beta) = \sum_{i=1}^N \log P(y_i | x_i, \beta)$$

- We want to find the value of β that leads to the highest value of the conditional log likelihood:

$$\ell(\beta) = \sum_{i=1}^N \log P(y_i \mid x_i, \beta)$$

We want to find the values of β that make the value of this function the greatest

$$\sum_{\langle x, y=+1 \rangle} \log P(1 \mid x, \beta) + \sum_{\langle x, y=0 \rangle} \log P(0 \mid x, \beta)$$

$$\frac{\partial}{\partial \beta_i} \ell(\beta) = \sum_{\langle x, y \rangle} (y - \hat{p}(x)) x_i$$

Gradient descent

Algorithm 1 Logistic regression gradient descent

- 1: Data: training data $x \in \mathbb{R}^F, y \in \{0, 1\}$
 - 2: $\beta = 0^F$
 - 3: **while** not converged **do**
 - 4: $\beta_{t+1} = \beta_t + \alpha \sum_{i=1}^N (y_i - \hat{p}(x_i)) x_i$
 - 5: **end while**
-

If y is 1 and $p(x) = 0$, then this still pushes the weights a lot

If y is 1 and $p(x) = 0.99$, then this still pushes the weights just a little bit

Stochastic gradient descent

- Batch gradient descent reasons over every training data point for each update of β . This can be slow to converge.
- Stochastic gradient descent updates β after each data point.

Algorithm 2 Logistic regression stochastic gradient descent

```
1: Data: training data  $x \in \mathbb{R}^F, y \in \{0, 1\}$ 
2:  $\beta = 0^F$ 
3: while not converged do
4:   for  $i = 1$  to  $N$  do
5:      $\beta_{t+1} = \beta_t + \alpha (y_i - \hat{p}(x_i)) x_i$ 
6:   end for
7: end while
```

Practicalities

- When calculating the $P(y | x)$ or in calculating the gradient, you don't need to loop through all features — only those with **nonzero** values
- (Which makes sparse, binary values useful)

$$P(y = 1 | x, \beta) = \frac{1}{1 + \exp\left(-\sum_{i=1}^F x_i \beta_i\right)}$$

$$\frac{\partial}{\partial \beta_i} \ell(\beta) = \sum_{\langle x, y \rangle} (y - \hat{p}(x)) x_i$$

Many features that show up rarely
may likely only appear (by chance)
with one label

More generally, may appear so few
times that the noise of randomness
dominates

β = coefficients

Feature	β
like	2.1
did not like	1.4
in_pos_dict_MPQA	1.7
in_neg_dict_MPQA	-2.1
in_pos_dict_LIWC	1.4
in_neg_dict_LIWC	-3.1
author=ebert	-1.7
author=ebert $\wedge$ dog $\wedge$ starts with "in"	30.1

Feature selection

- We could threshold features by minimum count but that also throws away information
- We can take a probabilistic approach and encode a prior belief that all β should be 0 unless we have strong evidence otherwise

L2 regularization

$$\ell(\beta) = \underbrace{\sum_{i=1}^N \log P(y_i | x_i, \beta)}_{\text{we want this to be high}} - \underbrace{\eta \sum_{j=1}^F \beta_j^2}_{\text{but we want this to be small}}$$

- We can do this by changing the function we're trying to optimize by adding a penalty for having values of β that are high
- This is equivalent to saying that each β element is drawn from a Normal distribution centered on 0.
- η controls how much of a penalty to pay for coefficients that are far from 0 (optimize on development data)

no L2 regularization

33.83	Won Bin
29.91	Alexander Beyer
24.78	Bloopers
23.01	Daniel Brühl
22.11	Ha Jeong-woo
20.49	Supernatural
18.91	Kristine DeBell
18.61	Eddie Murphy
18.33	Cher
18.18	Michael Douglas

some L2 regularization

2.17	Eddie Murphy
1.98	Tom Cruise
1.70	Tyler Perry
1.70	Michael Douglas
1.66	Robert Redford
1.66	Julia Roberts
1.64	Dance
1.63	Schwarzenegger
1.63	Lee Tergesen
1.62	Cher

high L2 regularization

0.41	Family Film
0.41	Thriller
0.36	Fantasy
0.32	Action
0.25	Buddy film
0.24	Adventure
0.20	Comp Animation
0.19	Animation
0.18	Science Fiction
0.18	Bruce Willis

L1 regularization

$$\ell(\beta) = \underbrace{\sum_{i=1}^N \log P(y_i | x_i, \beta)}_{\text{we want this to be high}} - \underbrace{\eta \sum_{j=1}^F |\beta_j|}_{\text{but we want this to be small}}$$

- L1 regularization encourages coefficients to be **exactly** 0.
- η again controls how much of a penalty to pay for coefficients that are far from 0 (optimize on development data)

Multiclass logistic regression

$$P(Y = y \mid X = x; \beta) = \frac{\exp(x^\top \beta_y)}{\sum_{y' \in \mathcal{Y}} \exp(x^\top \beta_{y'})}$$

output space

$$\mathcal{Y} = \{1, \dots, K\}$$

x = feature vector

β = coefficients

Feature	Value
the	0
and	0
bravest	0
love	0
loved	0
genius	0
not	0
fruit	1
<i>BIAS</i>	1

Feature	β_{positive}	β_{negative}	β_{neutral}
the	1.33	-0.80	-0.54
and	1.21	-1.73	-1.57
bravest	0.96	-0.05	0.24
love	1.49	0.53	1.01
loved	-0.52	-0.02	2.21
genius	0.98	0.77	1.53
not	-0.96	2.14	-0.71
fruit	0.59	-0.76	0.93
<i>BIAS</i>	-1.92	-0.70	0.94

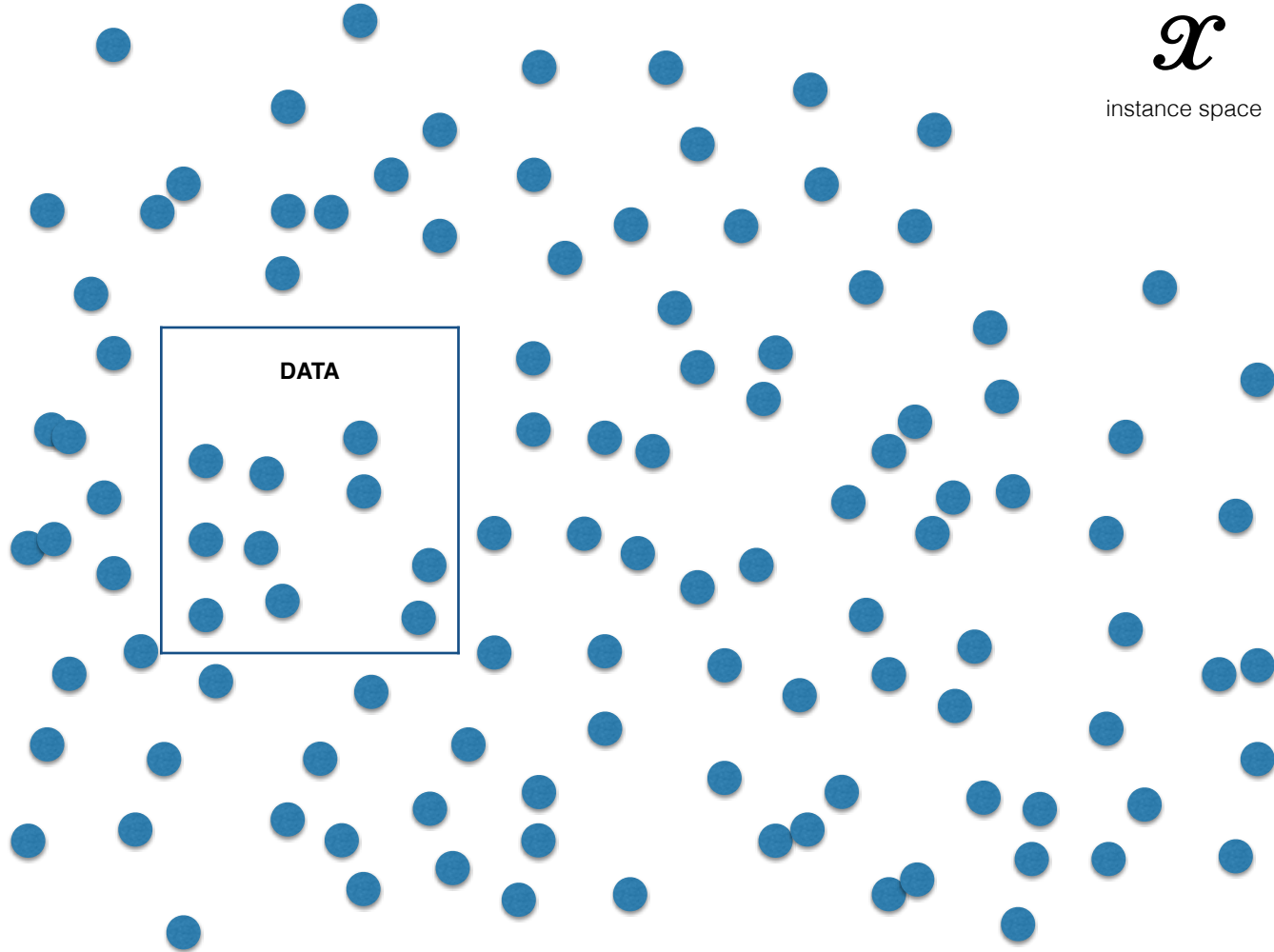
Note that we have 3 sets of coefficients here — one for each class (positive, negative, neutral)

Evaluation

- A critical part of development new algorithms and methods and demonstrating that they work

$\mathcal{X}$

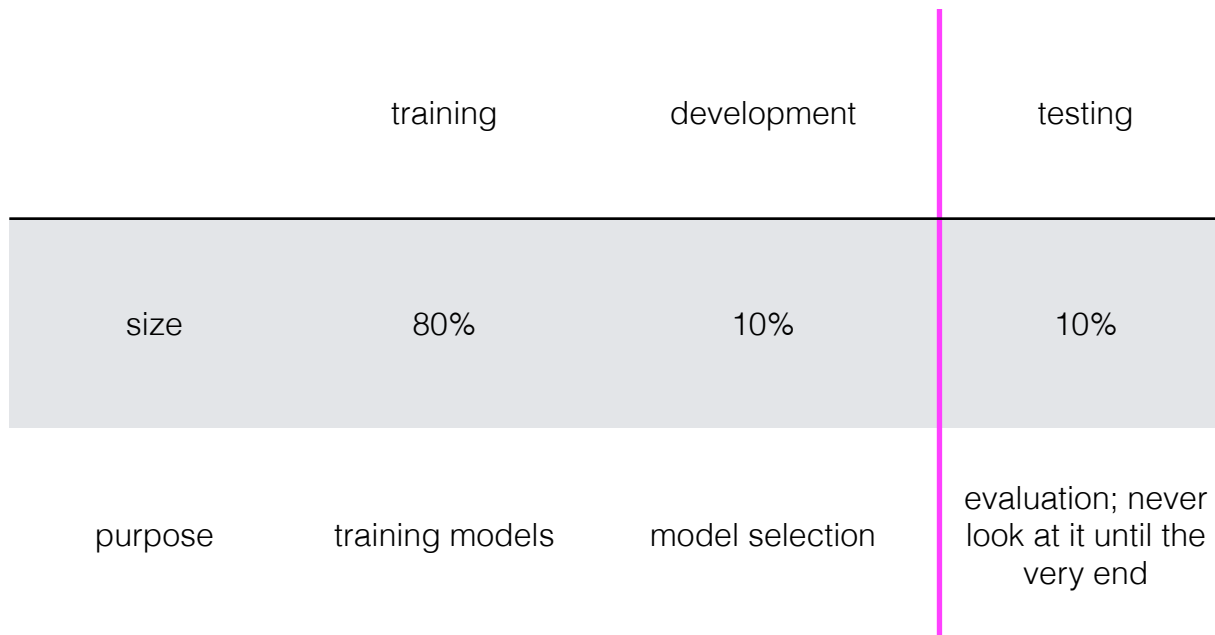
instance space



instance space

train	dev	test
		

Experiment design



Multiclass confusion matrix

		Predicted ($\hat{y}$)		
		Positive	Negative	Neutral
True (y)	Positive	100	2	15
	Negative	0	104	30
	Neutral	30	40	70

Accuracy

$$\frac{1}{N} \sum_{i=1}^N I[\hat{y}_i = y_i]$$

$$I[x] = \begin{cases} 1 & \text{if } x \text{ is true} \\ 0 & \text{otherwise} \end{cases}$$

		Predicted ($\hat{y}$)		
		Positive	Negative	Neutral
True (y)	Positive	100	2	15
	Negative	0	104	30
	Neutral	30	40	70

Precision

Precision(POS) =

$$\frac{\sum_{i=1}^N I(y_i = \hat{y}_i = \text{POS})}{\sum_{i=1}^N I(\hat{y}_i = \text{POS})}$$

Precision: proportion of predicted class that are actually that class.

		Predicted ($\hat{y}$)		
		Positive	Negative	Neutral
True (y)	Positive	100	2	15
	Negative	0	104	30
	Neutral	30	40	70

Recall

Recall(POS) =

$$\frac{\sum_{i=1}^N I(y_i = \hat{y}_i = \text{POS})}{\sum_{i=1}^N I(y_i = \text{POS})}$$

Predicted ($\hat{y}$)

Positive Negative Neutral

True (y)

Positive

100

2

15

Negative

0

104

30

Neutral

30

40

70

Recall: proportion of true class that are predicted to be that class.

F score

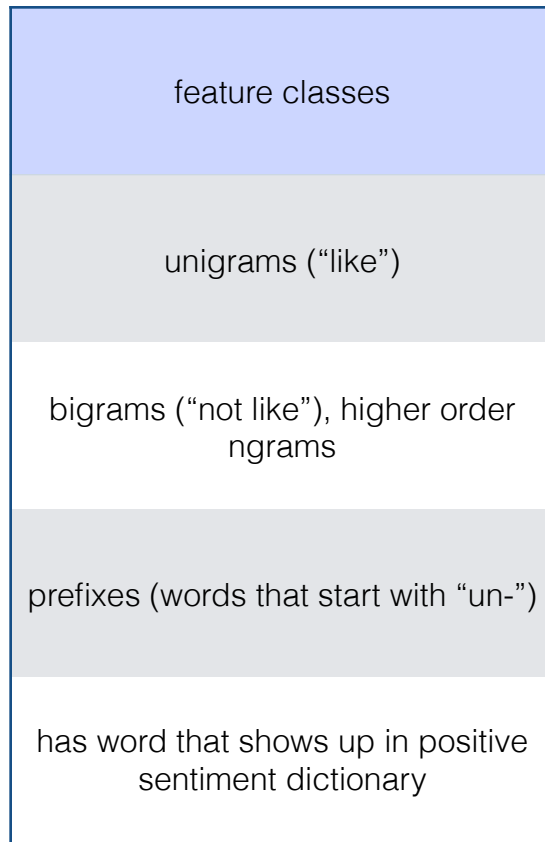
$$F = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

Majority class baseline

- Pick the label that occurs the most frequently in the training data. (Don't count the test data!)
- Predict that label for every data point in the test data.

Features

- Features are where you can encode your own **domain understanding** of the problem.



Features

Task	Features
Sentiment classification	Words, presence in sentiment dictionaries, etc.
Keyword extraction	
Fake news detection	
Authorship attribution	