

計算言語学

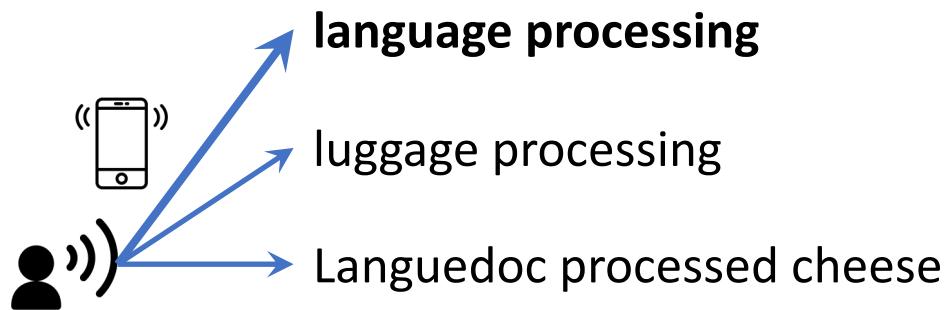
言語モデルと分類

東京大学生産技術研究所
吉永 直樹

site: <http://www.tkl.iis.u-tokyo.ac.jp/~ynaga/class/cl/>

言語の自然さをモデル化する

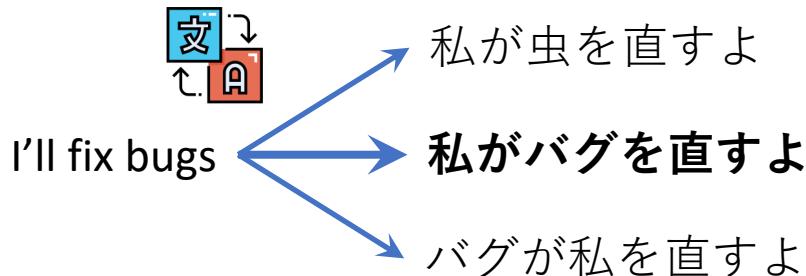
音声認識



予測入力



機械翻訳



誤り訂正

文の確立を求めるためい



文の確率を求めたい

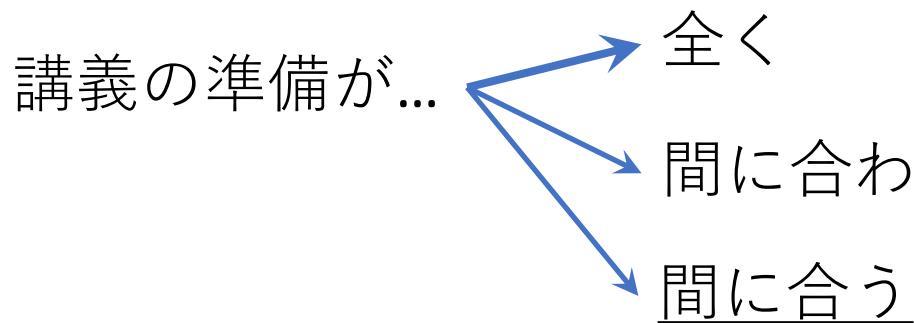
言語モデル

- 文の確率をモデル化

$$P(\text{“私がバグを直すよ”}) = 2.3 \cdot 10^{-5}$$

$$P(\text{“バグが私を直すよ”}) = 1.2 \cdot 10^{-12}$$

or 単語列(**n-gram**)の次の単語の確率をモデル化



$$P(\text{“間に合う”} | \text{“講義の準備が”}) = 7.8 \cdot 10^{-23}$$

コーパスを用いた文の確率の計算

- 素朴なアイデア: コーパス中の頻度で確率を計算

$$P(\textcolor{blue}{s}) = \frac{\text{文の頻度} C(s)}{\sum_{s'} C(s')}$$

$$P(\textcolor{blue}{w}|\textcolor{red}{h}) = \frac{\text{単語} C(hw)}{C(h)}$$

(直前の)単語列

悲叹の顔の絵文字 単語列が長くなるに従い、頻度は指数的に減少
(データスペースネス)

$$P(\text{“間に合う”} | \text{“講義の準備が”}) = \frac{C(\text{“講義の準備が間に合う”})}{C(\text{“講義の準備が”})}$$

頻度0 → 確率0 悲叹の顔の絵文字

マルコフ過程

- 確率の連鎖公式で文の確率を分解

$$\begin{aligned} P(w_1 \dots w_n) &= P(w_1)P(w_2 \dots w_n | w_1) \\ &= P(w_1)P(w_2 | w_1)P(w_3 \dots w_n | w_1^2) \\ &= \dots \\ &= P(w_1)P(w_2 | X_1)P(w_3 | w_1^2) \dots P(w_n | w_1^{n-1}) \\ &= \prod_{k=1}^n P(w_k | w_1^{k-1}) \end{aligned}$$

- 単語の出現が直前の N 語にしか依存しないという
(N 次)マルコフ過程を仮定し、上式を近似

$$P(w_k | w_1^{k-1}) \approx P(w_k | w_{\textcolor{red}{k-N+1}}^{k-1})$$
$$P(w_1 \dots w_n) \approx \prod_{k=1}^N P(w_k | w_{\textcolor{red}{k-N+1}}^{k-1}) \quad \text{講義の準備が間に合う}$$

N-gram モデル

- N-gram モデル

N 次マルコフ過程に基づく言語モデル

$$P(w_1^n) = \prod_{k=1}^n P(w_k | w_{k-N+1}^{k-1}) = \begin{cases} \prod_{k=1}^n P(w_k) & (N = 1, unigram) \\ \prod_{k=1}^n P(w_k | w_{k-2}^{k-1}) & (N = 2, bigram) \\ \prod_{k=1}^n P(w_k | w_{k-3}^{k-1}) & (N = 3, trigram) \end{cases}$$

- 文の確率は $N - 1$ 個の文頭記号($<\text{s}>$)と 1 個の文末記号($</\text{s}>$)を付与して計算

$N=3$ の場合 $P(<\text{s}> <\text{s}> \text{ 講義 の 準備 が 間に合わ ない } </\text{s}>)$

$$= P(\text{講義} | <\text{s}> <\text{s}>) P(\text{の} | <\text{s}> \text{ 講義})$$

$$P(\text{準備} | \text{講義 の}) P(\text{が} | \text{の準備})$$

$$P(\text{間に合わ} | \text{準備 が}) P(\text{ない} | \text{が 間に合わ})$$

$$P(</\text{s}> | \text{間に合わ ない})$$

最尤推定 (maximum likelihood estimation; MLE)

- (言語モデルによる) コーパスの条件付き確率(尤度)を最大化するように確率を推定
 - コーパスにおける単語列の頻度を用いて n-gram の確率を計算 (n-gram の頻度を正規化(相対頻度))

$$P(w_k | w_{k-N+1}^{k-1}) = \frac{C(w_{k-N+1}^{k-1} w_k)}{\sum_w C(w_{k-N+1}^{k-1} w)} = \frac{C(w_{k-N+1}^k)}{C(w_{k-N+1}^{k-1})}$$

- 通常、桁落ちを避けるため確率を対数にして計算
(対数尤度)

$$\log P(w_1^n) = \log \prod_{k=1}^n P(w_k | w_{k-N+1}^{k-1}) = \sum_{k=1}^n \log P(w_k | w_{k-N+1}^{k-1})$$

(言語モデルの)自動評価

- **Intrinsic** な評価
当該タスクに閉じた
直接的評価



低成本, 再現性大



間接的

- **Extrinsic** な評価
応用(例: 機械翻訳)における
性能改善等を評価

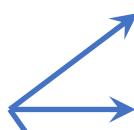


本質的



高コスト, 再現性小

I played a black bass.



私は黒いベースを引いた

私は黒いベースを演奏した

私はブラックバスを演じた



Intrinsic 必須, Extrinsic は可能ならという場合が多い

学習・開発・テストコーカス

- コーカスを以下の3つに**排他的**に分けて学習・評価
 - 学習データ (training data)
モデルの学習(パラメタの推定)を使う
 - 開発データ (development data)
学習データから得たモデルの性能の確認に使う (開発)
(モデルのハイパーパラメタの調整など)
 - テストデータ (test data)
学習データから得たモデルの性能の最終評価に使う

テストデータは実際の応用を想定し、未知データとして扱うべきでモデルの開発に利用するのは NG (closed test)

言語モデルの評価尺度: (テストセット)パープレキシティ

- 直感:
良い言語モデル = (未知の)テキストを高確率で生成
- (テストセット)パープレキシティ

$$PP(w_1^n) = P(w_1^n)^{-\frac{1}{n}}$$

$$\begin{aligned} &= \sqrt[n]{\prod_{k=1}^n \frac{1}{P(w_k | w_1^{k-1})}} \\ &= \sqrt[n]{\prod_{k=1}^n \frac{1}{P(w_k | w_{k-N+1}^{k-1})}} \quad \left\{ \begin{array}{ll} \sqrt[n]{\prod_{k=1}^n \frac{1}{P(w_k)}} & \text{unigram} \\ \sqrt[n]{\prod_{k=1}^n \frac{1}{P(w_k | w_{k-2}^{k-1})}} & \text{bigram} \end{array} \right. \\ &\qquad\qquad\qquad (N=1) \\ &\qquad\qquad\qquad (N=2) \end{aligned}$$

パープレキシティの解釈

- 与えられた単語列の次の単語を予測するときの重み付き平均分岐数 (何択問題を解いているか)

例) ランダム数字列に対する unigram モデル

$$\begin{aligned} P(w_1^n) &= \sqrt[n]{\prod_{k=1}^n \frac{1}{P(w_k)}} \\ &= \sqrt[n]{\prod_{k=1}^n \frac{1}{\frac{1}{10}}} \\ &= 10 \end{aligned}$$

例) Wall Street Journal での言語モデルのパープレキシティ

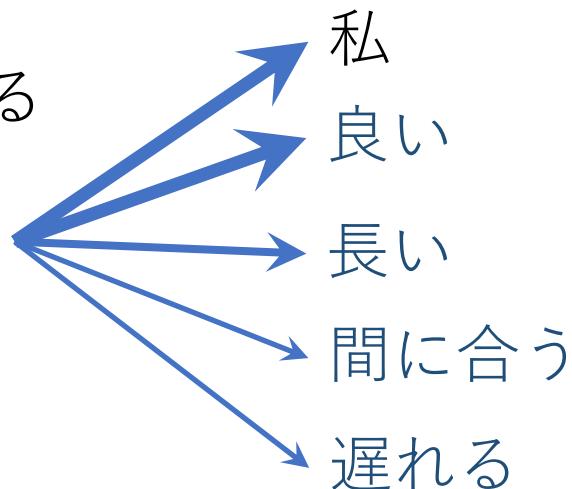
	Unigram	Bigram	Trigram
Perplexity	962	170	109

ゼロ頻度問題

- モデルの学習に用いる学習データ(または言語資源)にない単語(未知語)やn-gramを扱うことができない
 - 未知語: 新語 + 低頻度語
 - モデルパラメタを増やすに従い深刻化

例) N-gram モデルで予測性能を
上げるため N を増やしてみる

計算言語学の講義の準備が...

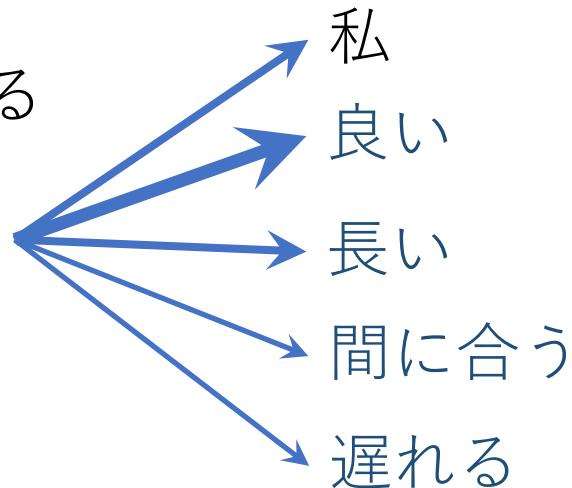


ゼロ頻度問題

- モデルの学習に用いる学習データ(または言語資源)にない単語(未知語)やn-gramを扱うことができない
 - 未知語: 新語 + 低頻度語
 - モデルパラメタを増やすに従い深刻化

例) N-gram モデルで予測性能を
上げるため N を増やしてみる

計算言語学の講義の準備が...

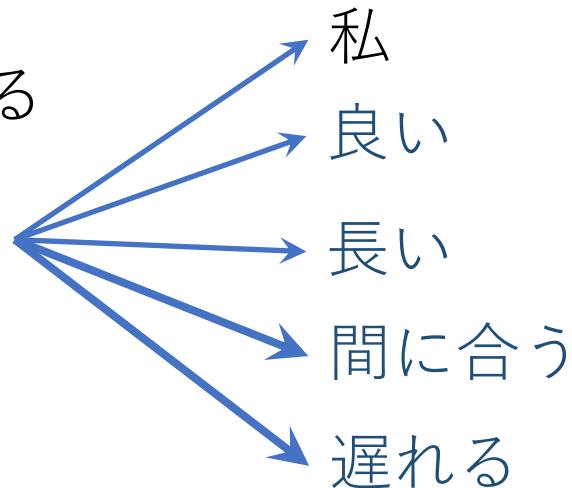


ゼロ頻度問題

- モデルの学習に用いる学習データ(または言語資源)にない単語(未知語)やn-gramを扱うことができない
 - 未知語: 新語 + 低頻度語
 - モデルパラメタを増やすに従い深刻化

例) N-gram モデルで予測性能を
上げるため N を増やしてみる

計算言語学の講義の準備が...

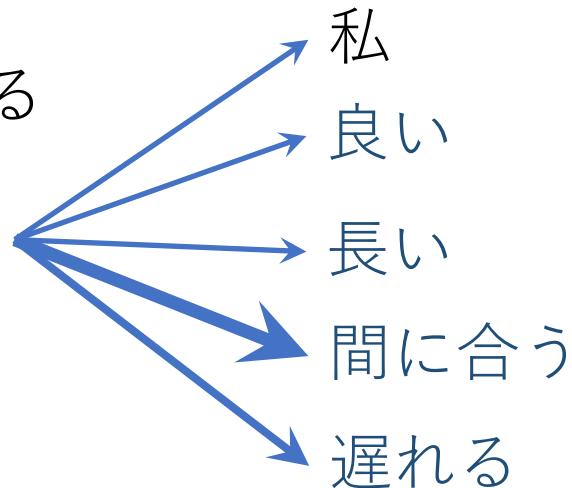


ゼロ頻度問題

- モデルの学習に用いる学習データ(または言語資源)にない単語(未知語)やn-gramを扱うことができない
 - 未知語: 新語 + 低頻度語
 - モデルパラメタを増やすに従い深刻化

例) N-gram モデルで予測性能を
上げるため N を増やしてみる

計算言語学の講義の準備が...

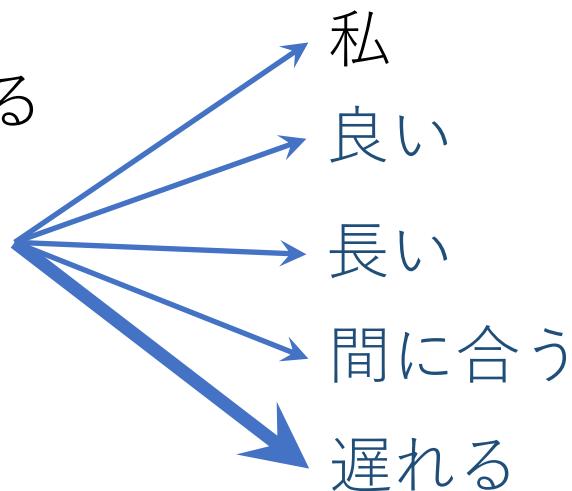


ゼロ頻度問題

- モデルの学習に用いる学習データ(または言語資源)にない単語(未知語)やn-gramを扱うことができない
 - 未知語: 新語 + 低頻度語
 - モデルパラメタを増やすに従い深刻化

例) N-gram モデルで予測性能を
上げるため N を増やしてみる

計算言語学の講義の準備が...



$$P(\text{“遅れる”} | \text{“計算言語学の講義の準備が”}) = \frac{C(\text{“計算言語学の講義の準備が遅れる”})}{C(\text{“計算言語学の講義の準備が”})}$$
$$= 0$$

未知語処理

- コーパス中の一部の単語を未知語とみなして処理
 - 学習コーパス中で未知語と同一視する語を決めて、事前に特殊な単語 (<UNK>) に置き換えて語彙に含めて学習
 - テストコーパスを評価する際は、モデルの語彙にない語は <UNK> に置き換えて処理
 - どのように未知語として扱う語を決めるか？
 - 事前に用意した語彙集合に含まれない単語全て
 - コーパス中で一定頻度以下の語

未知語は全て同じような振る舞いをすることが期待される
- 注) 語彙の集合が異なるとパープレキシティは比較不可

未知 n-gram への対応

- 未知 n-gram (既知語から構成される未知の n-gram)
- 対策1: スムージング (ディスカウンティング)
 - 観測された n-gram の頻度を減らし、観測されていない n-gram に頻度を分配する
- 対策2: 線形補間
 - 低次 n-gram との重み付き和で n-gram の確率を推定
- 対策3: バックオフ
 - 低次 n-gram の確率から n-gram の確率を推定

ラプラススムージング

- 全ての n-gram の頻度に 1 を追加して確率を計算

$$P(w_i) = \frac{C(w_i)}{\sum_w C(w)} \quad \rightarrow \quad P_{\text{Laplace}}(w_i) = \frac{C(w_i) + 1}{\sum_w (C(w) + 1)}$$

unigram の場合

$$= \frac{C(w_n) + 1}{N + |V|}$$

1-gram の総出現数
(コーパスサイズ+1)

単語の種類数
(語彙数)

- 観測された n-gram の頻度を減らして未知 n-gram に分配

$$C^*(w_i) = C(w_i) \frac{P_{\text{Laplace}}(w_i)}{P(w_i)} = (C(w_i) + 1) \frac{N}{N + |V|}$$

頻度の総和は不变

$$\sum_{w_i} (C(w_i) - C^*(w_i)) = \sum_{w_i} \left(C(w_i) - (C(w_i) + 1) \frac{N}{N + |V|} \right)$$

$$= \sum_{w_i} \frac{C(w_i)|V| - N}{N + |V|}$$
$$= 0$$

ラプラスムージング: 計算例

- Bigram の場合

$$P_{\text{Laplace}}(w_n | w_{n-1}) = \frac{C(w_{n-1}w_n) + 1}{\sum_w (C(w_{n-1}w) + 1)} = \frac{C(w_{n-1}w_n) + 1}{N + |V|}$$

2-gramの総出現数
(コーパスサイズ+1)

単語の種類数
(語彙数)

Berkeley Restaurant Project Corpus (9332 sent.)

	i	want	to	eat	chinese	food	lunch	speed
i	5	827	0	9	0	0	0	2
want	2	0	608	1	6	6	5	1
to	2	0	4	686	2	0	6	211
eat	0	0	2	0	16	2	42	0
chinese	1	0	0	0	0	82	1	0
food	15	0	15	0	1	4	0	0
lunch	2	0	0	0	0	1	0	0
speed	1	0	1	0	0	0	0	0

ラプラスムージング (2/2)

- Bigram の場合

$$P_{\text{Laplace}}(w_n | w_{n-1}) = \frac{C(w_{n-1}w_n) + 1}{\sum_w (C(w_{n-1}w) + 1)} = \frac{C(w_{n-1}w_n) + 1}{N + |V|}$$

2-gramの総出現数
(コーパスサイズ+1)

単語の種類数
(語彙数)

Berkeley Restaurant Project Corpus (9332 sent.)

	i	want	to	eat	chinese	food	lunch	speed
i	6	828	1	10	1	1	1	3
want	3	1	609	2	7	7	6	2
to	3	1	5	687	3	1	7	212
eat	1	1	3	1	17	3	43	1
chinese	2	1	1	1	1	83	2	1
food	16	1	16	1	2	5	1	1
lunch	3	1	1	1	1	2	1	1
speed	2	1	2	1	1	1	1	1

ラプラスムージング (2/2)

- Bigram の場合

$$P_{\text{Laplace}}(w_n | w_{n-1}) = \frac{C(w_{n-1}w_n) + 1}{\sum_w (C(w_{n-1}w) + 1)} = \frac{C(w_{n-1}w_n) + 1}{N + |V|}$$

2-gramの総出現数
(コーパスサイズ+1)

単語の種類数
(語彙数)

Berkeley Restaurant Project Corpus (9332 sent.)

	i	want	to	eat	chinese	food	lunch	speed
i	0.0015	0.21	0.00025	0.0025	0.00025	0.00025	0.00025	0.00075
want	0.0013	0.00042	0.26	0.00084	0.0029	0.0029	0.0025	0.00084
to	0.00078	0.00026	0.0013	0.18	0.00078	0.00026	0.0018	0.055
eat	0.00046	0.00046	0.0014	0.00046	0.0078	0.0014	0.02	0.00046
chinese	0.0012	0.00062	0.00062	0.00062	0.00062	0.052	0.0012	0.00062
food	0.0063	0.00039	0.0063	0.00039	0.00079	0.002	0.00039	0.00039
lunch	0.0017	0.00056	0.00056	0.00056	0.00056	0.0011	0.00056	0.00056
speed	0.0012	0.00058	0.0012	0.00058	0.00058	0.00058	0.00058	0.00058

Add-k スムージング

- 問題: ラプラススムージングで頻度に 1 を決め打ちで足すのは多過ぎる
- 全ての n-gram の頻度に k を追加して確率を計算
 - k はハイパーパラメタ (開発データで調整)

$$P_{\text{Add-}k}(w_n | w_{n-1}) = \frac{C(w_{n-1} w_n) + k}{\sum_w (C(w_{n-1} w) + k)} = \frac{C(w_{n-1} w_n) + k}{N + k|V|}$$

- 文書分類等ではうまく動くが、言語モデルでは不十分との報告 [Gale+ 1994]

線形補間

- 低次 n-gram との重み付き和で n-gram の確率を計算

trigram の場合

$$\begin{aligned}\hat{P}(w_n | w_{n-2} w_{n-1}) &= \lambda_1 P(w_n | w_{n-2} w_{n-1}) \\ &\quad + \lambda_2 P(w_n | w_{n-1}) \\ &\quad + \lambda_3 P(w_n) \\ \sum_i \lambda_i &= 1\end{aligned}$$

- 補完係数は開発データで推定 (EMで尤度最大化)
- 補完係数を履歴ごとに別に用意するバージョン

$$\begin{aligned}\hat{P}(w_n | w_{n-2} w_{n-1}) &= \lambda_1(w_{n-2}^{n-1}) P(w_n | w_{n-2} w_{n-1}) \\ &\quad + \lambda_2(w_{n-2}^{n-1}) P(w_n | w_{n-1}) \\ &\quad + \lambda_3(w_{n-2}^{n-1}) P(w_n) \\ \sum_i \lambda_i(w_{n-2}^{n-1}) &= 1\end{aligned}$$

バックオフ

- 未知 n -gram の確率のみ低次 n -gram の確率から推定

$$P_{\text{BO}}(w_n | w_{n-N+1}^{n-1}) = \begin{cases} P^*(w_n | w_{n-N+1}^{n-1}) & \text{if } C(w_{n-N+1}^n) > 0 \\ \alpha(w_{n-N+1}^{n-1}) P_{\text{BO}}(w_n | w_{n-N+2}^{n-1}) & \text{otherwise} \end{cases}$$

- P^*, α は確率の制約を満たすようスケーリング

超巨大な n-gram 言語モデルを扱う場合 (1/2)

- Google n-gram: 1,024,908,267,229 語 → 1-5 n-gram
- データサイズへの対応と活用:
 - 単語列: 直接文字列として保存せずハッシュ値で表現
 - Bloom filter の亜種で近似的に確率に割当
 - 確率(バックオフ値): float (4-bytes) → 4-8 bits (量子化)
 - 言語モデル: Stupid バックオフ [Brants+ 2007]

観測された n-gram
は最尤推定

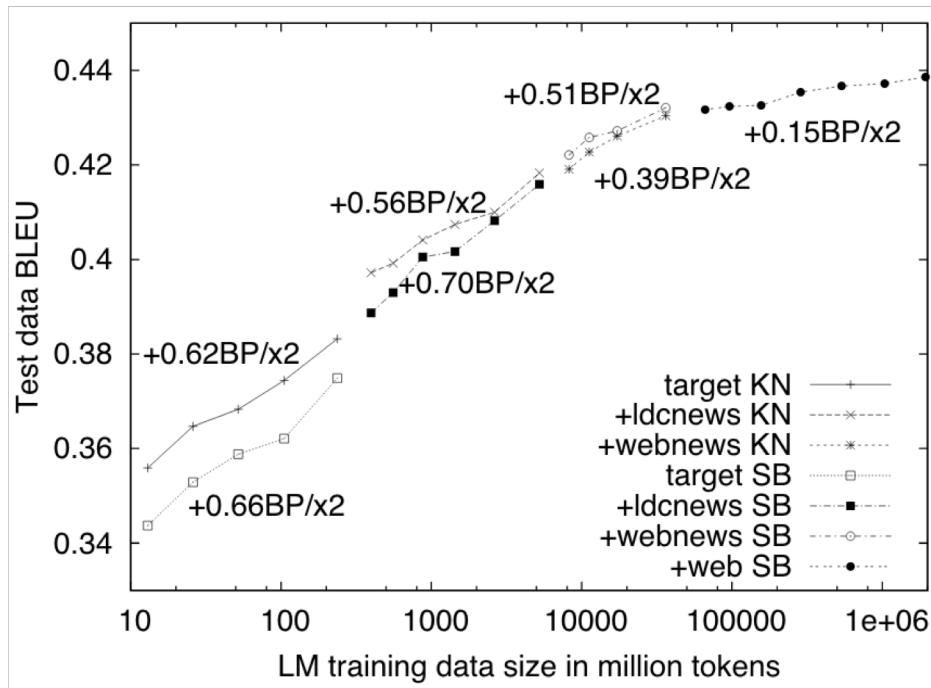
$$S(w_n | w_{n-N+1}^{n-1}) = \begin{cases} \frac{C(w_{n-N+1}^n)}{C(w_{n-N+1}^{n-1})} & \text{if } C(w_{n-N+1}^n) > 0 \\ \alpha S(w_n | w_{n-N+2}^{n-1}) & \text{otherwise} \end{cases}$$

もはや確率ではないことに注意

未知 n-gram は文脈非依存の α
でバックオフ (0.4 で大体 ok)

超巨大な n-gram 言語モデルを扱う場合 (1/2)

- Stupid バックオフの **extrinsic** な評価
 - アラビア語から英語への翻訳タスク



from [Brants+ 2007, Figure 5]

コーパスのサイズを大きくすればするほど、
スムージング/バックオフ手法間の差は減少

言語をカテゴリ(クラス)に分ける

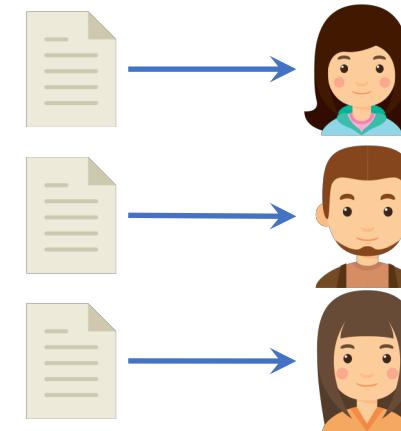
評判分析

欠点が少ない → 肯定的

野菜が少ない → 否定的

ラベンダーの香り → 肯定的

著者分類



文書分類

→ スポーツ

→ 政治

→ 科学

品詞解析

Lucy in the sky with diamonds
NNP IN DT NN IN NNS

Lucy in the Sky with Diamonds
NNP NNP NNP NNP NNP



相互依存あり → 構造分類

教師あり学習に基づく分類器

- タスク: 入力 x をクラス y に分類する
 - 例) スパム分析: テキスト → ham or spam

- 生成モデル: 同時確率 $P(x, y)$ をモデル化

$$\begin{aligned}\hat{y} &= \operatorname{argmax}_{y \in Y} P(y|x) = \operatorname{argmax}_{y \in Y} \frac{P(x|y)P(y)}{P(y)} \\ &= \operatorname{argmax}_{y \in Y} \underbrace{P(x|y)}_{\text{尤度}} \underbrace{P(y)}_{\text{事前確率}}\end{aligned}$$

- 識別モデル: 条件付き確率 $P(y|x)$ を直接モデル化

$$\hat{y} = \operatorname{argmax}_{y \in Y} P(y|x)$$

教師あり学習に基づく分類器

- タスク: 入力 x をクラス y に分類する
 - 例) スパム分析: テキスト → ham or spam

- 生成モデル: 同時確率 $P(x, y)$ をモデル化

$$\begin{aligned}\hat{y} &= \operatorname{argmax}_{y \in Y} P(y|x) = \operatorname{argmax}_{y \in Y} \frac{P(x|y)P(y)}{P(y)} \\ &= \operatorname{argmax}_{y \in Y} \underbrace{P(x|y)}_{\text{尤度}} \underbrace{P(y)}_{\text{事前確率}}\end{aligned}$$

- 識別モデル: 条件付き確率 $P(y|x)$ をモデル化

$$\hat{y} = \operatorname{argmax}_{y \in Y} P(y|x)$$

Naïve Bayes 分類器

- **Naïve Bayes**

入力のベクトル表現の要素(素性)間に独立性の仮定を置いた生成モデルに基づく多値分類器

- 注) 素性は NLP 以外では特徴量と呼ばれることが多い

例) 文書分類: 文書 $d \rightarrow$ カテゴリ c 素性関数

$$\begin{aligned}\hat{c} &= \operatorname{argmax}_{c \in C} P(d|c)P(c) \quad \underbrace{\phi(d)}_{\text{素性関数}} = \{f_1, f_2, \dots, f_n\} \\ &= \operatorname{argmax}_{c \in C} P(f_1, f_2, \dots, f_n|c)P(c) \\ &= \operatorname{argmax}_{c \in C} P(f_1|c)P(f_2|c)\dots P(f_n|c)P(c) \\ &= \operatorname{argmax}_{c \in C} P(c) + \prod_{f \in F} P(f|c)\end{aligned}$$

Bag of words: 単純な素性関数

- Bag of words: 単語列を単語の集合と考え
(順序や構造を無視して)ベクトル化

d

I love this movie! It's sweet,
but with satirical humor.
The dialogue is great and the
adventure scenes are fun...



$\phi(d)$

the 2
fun 1
sweet 1
statirical 1
adventure 1
humor 1
but 1
are 1
I 1
... ...

$$\begin{pmatrix} 2 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ \dots \end{pmatrix}$$

- Naïve Bayes + Bag of words

$$\hat{c} = \operatorname{argmax}_{c \in C} P(d = w_1^n | c) P(c) = \operatorname{argmax}_{c \in C} P(c) \prod_i^n P(w_i | c)$$

Naïve Bayes 分類器の学習

- 正解コーパスに基づく**最尤推定**を使う

$$\hat{c} = \operatorname{argmax}_{c \in C} P(d = w_1^n | c) = \operatorname{argmax}_{c \in C} P(c) \prod_i^n P(w_i | c)$$

- 事前確率: クラス c の生成され易さ $\hat{P}(c) = \frac{N_c}{\sum_c N_c}$
 クラス c の文書数 全文書数
- 尤度: クラス c での各語の生成され易さ $\hat{P}(w_i | c) = \frac{C(w_i, c)}{\sum_{w \in V} C(w, c)}$

実用上の注意点:

- ゼロ頻度問題対策: 未知語処理・スムージング
- 桁落ちを防ぐため言語モデルと同様に**対数確率**を用いる

$$\hat{c} = \operatorname{argmax}_{c \in C} P(c) \prod_i P(w_i | c) = \operatorname{argmax}_{c \in C} \log P(c) + \sum_i \log P(w_i | c)$$

多クラス・多ラベル分類

- Multi-class (**one-of**) classification
 - 入力に対し、多数のラベルから一つを出力
 - ナイーブベイズでは(一応)自然に扱える
 - **one-versus-the-rest法**: ラベルごとに(そのラベルか否かの)2値分類器を学習し、確率(スコア)最大のラベルを出力
- Multi-label (**any-of**) classification
 - 入力に対し、多数のラベルから一つまたは複数を出力
 - ラベルごとに2値分類器を独立に学習・適用
 - 計算言語学・自然言語処理のタスクではあまりない

Complement Naïve Bayes [Rennie+ 2003]

- クラスの分布間の偏りが大きいときや超多クラスのときに頑健に動作するNaive Bayesの拡張
 - 対象クラス以外で素性が出現する確率を利用する

$$\begin{aligned}\hat{c} &= \operatorname{argmax}_{c \in C} P(d = w_1^n | c) = \operatorname{argmax}_{c \in C} P(c) \prod_i P(w_i | c) \\ &= \operatorname{argmax}_{c \in C} P(c) \prod_i \frac{1}{P(w_i | \bar{c})}\end{aligned}$$

c が低頻度だと
信頼性低い

c が低頻度でも
信頼性高い

教師あり学習に基づく分類器(再掲)

- タスク: 入力 x を適切なクラス y に分類する

- 例) スパム分析: テキスト → ham or spam

- 生成モデル: 同時確率 $P(x, y)$ をモデル化

$$\begin{aligned}\hat{y} &= \operatorname{argmax}_{y \in Y} P(y|x) = \operatorname{argmax}_{y \in Y} \frac{P(x|y)P(y)}{P(y)} \\ &= \operatorname{argmax}_{y \in Y} \underbrace{P(x|y)}_{\text{尤度}} \underbrace{P(y)}_{\text{事前確率}}\end{aligned}$$

- 識別モデル: 条件付き確率 $P(y|x)$ を直接モデル化

$$\hat{y} = \operatorname{argmax}_{y \in Y} P(y|x)$$

線形分類器

- 線形分類器 $y = \text{sgn}(\mathbf{w}^T \phi(x)) = \begin{cases} +1 & (\mathbf{w}^T \phi(x) \geq 0) \\ -1 & (\mathbf{w}^T \phi(x) < 0) \end{cases}$
- 入力を表現する素性ベクトル $\phi(x)$ に対しラベルを出力
- 学習**: 学習データから重みベクトル \mathbf{w} を推定
- 分類**: 分離超平面からの距離(マージン; $\mathbf{w}^T \phi(x)$)で分類

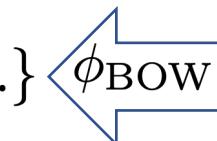
例) スパム分類(+1: ham; -1: spam)

学習データ: ( +1,  -1,  -1,  +1,  +1, ...)

x が “winner” を含む回数

$$\phi_{\text{BOW}}(x) = \{1, 0, 0, 1, 0, 1, 0, \dots\}$$

x が “money” を含む回数

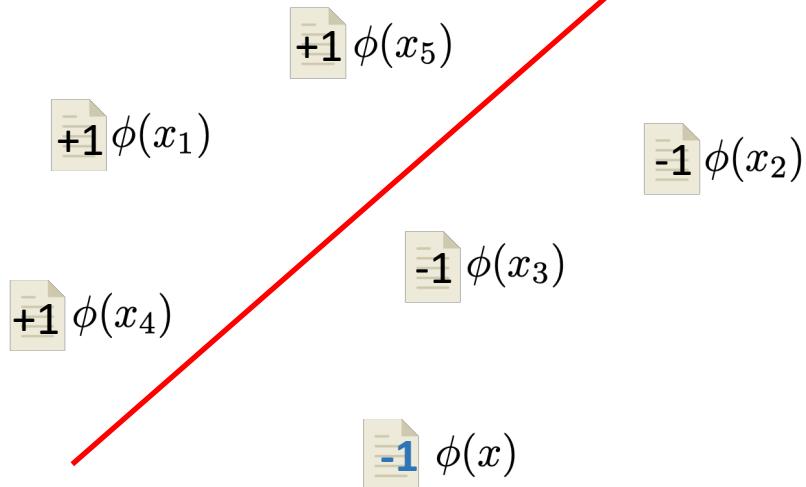


x


線形分類器

- 線形分類器 $y = \text{sgn}(\mathbf{w}^T \phi(x)) = \begin{cases} +1 & (\mathbf{w}^T \phi(x) \geq 0) \\ -1 & (\mathbf{w}^T \phi(x) < 0) \end{cases}$
- 入力を表現する素性ベクトル $\phi(x)$ に対しラベルを出力
- **学習**: 学習データから重みベクトル \mathbf{w} を推定
- **分類**: 分離超平面からの距離(マージン; $\mathbf{w}^T \phi(x)$)で分類

$$\mathbf{w}^T \phi(x) \geq 0 \quad (y = +1)$$



$$\text{分離超平面 } \mathbf{w}^T \phi(x) = 0$$

$\forall x, \phi(x)_0 = 1$ として w_0 にクラスの偏り
を学習させることもある(バイアス項)

素性工学

- 識別モデルでは素性間の独立性を仮定しないため素性関数の設計自由度が高い
- 素性関数をどう設計するか?
 - 基本素性: 単語表層, 品詞, 意味カテゴリ(上位語) etc.
 - 組合せ素性: 基本素性の組合せ
ゼロ頻度問題対処のため抽象的な基本素性の組合せが有効
- 深層学習では表現学習によりタスク毎に自動獲得
 - 高精度が達成できる反面, **ablation test** 等によるモデルの振る舞いの分析が困難(モデルの解釈性が低い)

Perceptron [Rosenblatt 1958]

- Perceptron: 誤り駆動の線形分類器
 - 訓練例を誤分類したらパラメタ更新
 - 一定回数更新後終了
 - 線形分離可能なら有限回で収束

$t = 1$

$$\mathbf{w}_1^T \phi(x) = 0$$

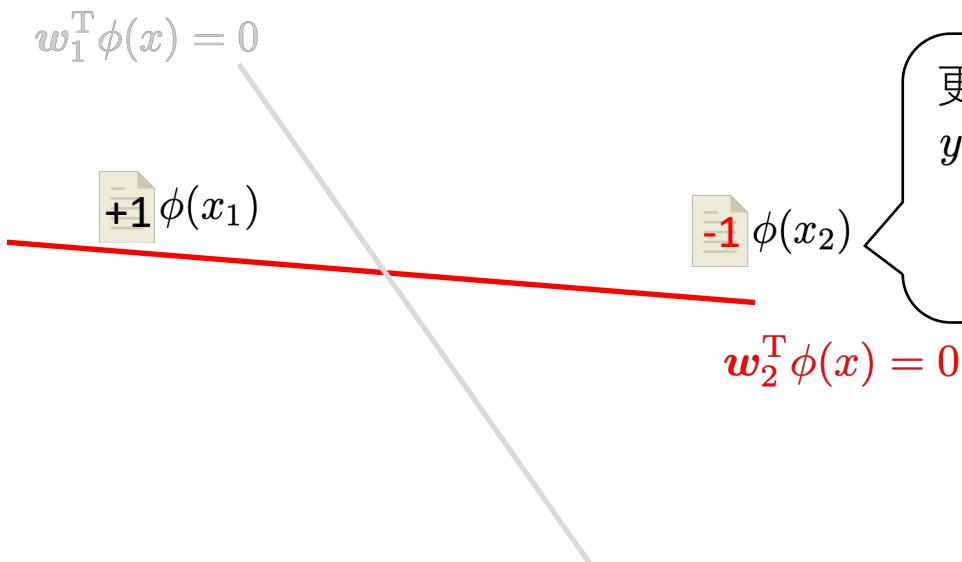

$$+1 \phi(x_1)$$

```
 $\mathbf{w}_0 = 0$ 
for  $t = 1$  to  $T$  do
    if  $y_t \mathbf{w}_{t-1}^T \phi(x_t) \leq 0$ 
         $\mathbf{w}_t = \mathbf{w}_{t-1} + y_t \mathbf{w}_{t-1}^T \phi(x_t)$ 
return  $\mathbf{w}_T$ 
```

Perceptron [Rosenblatt 1958]

- Perceptron: 誤り駆動の線形分類器
 - 訓練例を誤分類したらパラメタ更新
 - 一定回数更新後終了
 - 線形分離可能なら有限回で収束

$t = 2$



```
 $w_0 = 0$ 
for  $t = 1$  to  $T$  do
    if  $y_t w_{t-1}^T \phi(x_t) \leq 0$ 
         $w_t = w_{t-1} + y_t w_{t-1}^T \phi(x_t)$ 
return  $w_T$ 
```

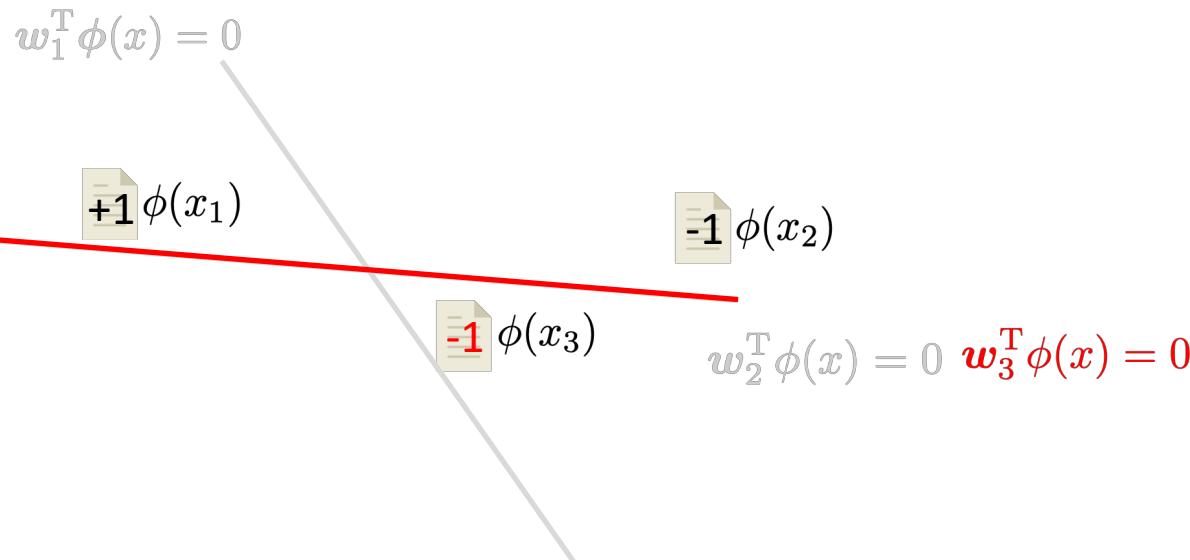
更新後は誤分類されにくくなっている

$$y_t w_t^T \phi(x_t) = y_t (w_{t-1}^T + y_t w_{t-1}^T \phi(x_t)) \phi(x_t)$$
$$= y_t w^T \phi(x_t) + \frac{\phi(x_t)^2}{2}$$
$$\leq 0 \quad \geq 0$$

Perceptron [Rosenblatt 1958]

- Perceptron: 誤り駆動の線形分類器
 - 訓練例を誤分類したらパラメタ更新
 - 一定回数更新後終了
 - 線形分離可能なら有限回で収束

$t = 3$

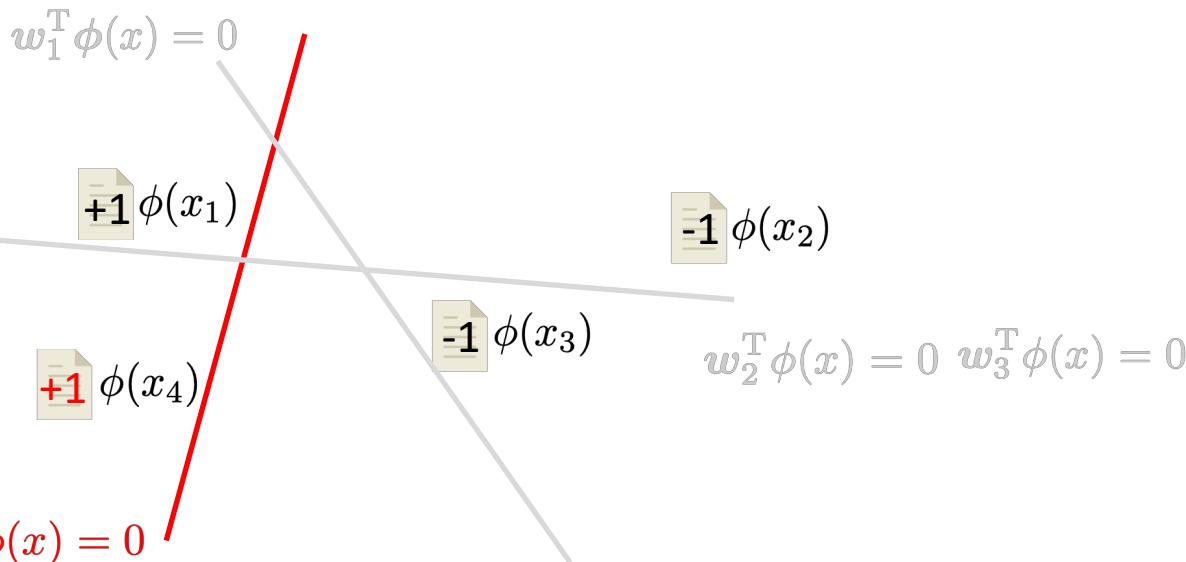


```
 $w_0 = 0$ 
for  $t = 1$  to  $T$  do
    if  $y_t w_{t-1}^T \phi(x_t) \leq 0$ 
         $w_t = w_{t-1} + y_t w_{t-1}^T \phi(x_t)$ 
return  $w_T$ 
```

Perceptron [Rosenblatt 1958]

- Perceptron: 誤り駆動の線形分類器
 - 訓練例を誤分類したらパラメタ更新
 - 一定回数更新後終了
 - 線形分離可能なら有限回で収束

$t = 4$

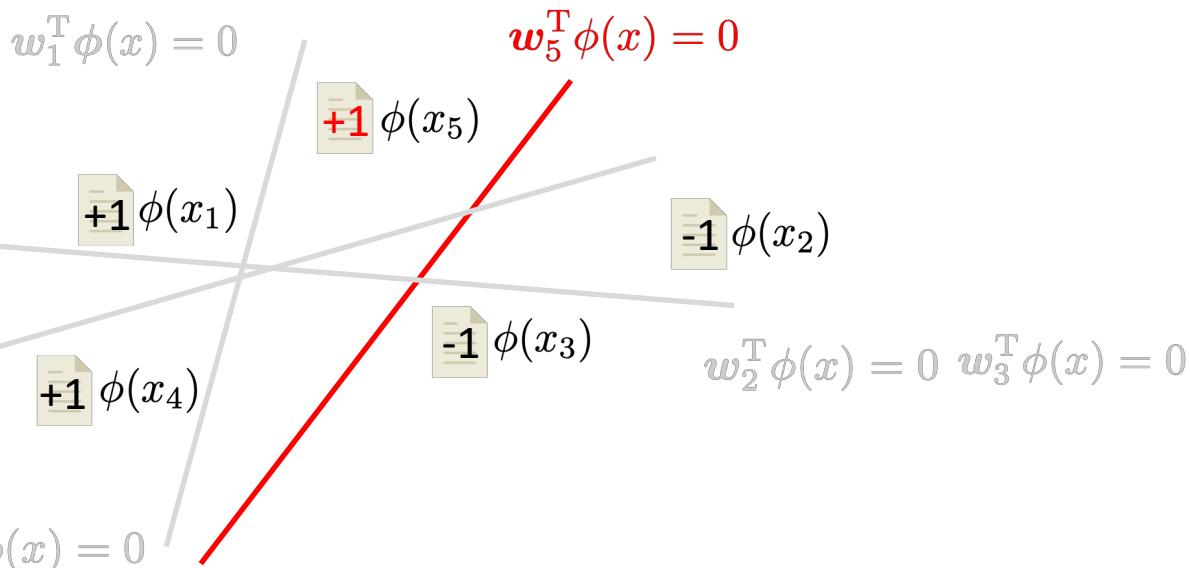


```
 $\mathbf{w}_0 = 0$ 
for  $t = 1$  to  $T$  do
    if  $y_t \mathbf{w}_{t-1}^T \phi(x_t) \leq 0$ 
         $\mathbf{w}_t = \mathbf{w}_{t-1} + y_t \mathbf{w}_{t-1}^T \phi(x_t)$ 
return  $\mathbf{w}_T$ 
```

Perceptron [Rosenblatt 1958]

- Perceptron: 誤り駆動の線形分類器
 - 訓練例を誤分類したらパラメタ更新
 - 一定回数更新後終了
 - 線形分離可能なら有限回で収束
 - 学習結果は入力順に大きく依存

$t = 5$



```
 $\mathbf{w}_0 = 0$ 
for  $t = 1$  to  $T$  do
    if  $y_t \mathbf{w}_{t-1}^T \phi(x_t) \leq 0$ 
         $\mathbf{w}_t = \mathbf{w}_{t-1} + y_t \mathbf{w}_{t-1}^T \phi(x_t)$ 
return  $\mathbf{w}_T$ 
```

Perceptron の収束性 [Novikoff 1962]

- $y(\mathbf{w}^T \phi(x_i)) \geq \gamma$ なる $\hat{\mathbf{w}} (\|\hat{\mathbf{w}}\| = 1)$ が存在する場合、誤文類する回数 t は $R = \max_{x_i} \|\phi(x_i)\|$ として高々

$$t \leq \frac{R^2}{\gamma^2}$$

- 証明

$$\begin{aligned}\|\mathbf{w}_t\|^2 &= \|\mathbf{w}_{t-1} + y_t \phi(x_t)\|^2 \leq \|\mathbf{w}_{t-1}\|^2 + \|\phi(x_t)\|^2 \\ &\leq \|\mathbf{w}_{t-1}\|^2 + R^2 \\ &\leq \dots \\ &\leq \|\mathbf{w}_0\|^2 + tR^2 \\ &= tR^2\end{aligned}$$

$$\begin{aligned}\hat{\mathbf{w}}^T \mathbf{w}_t &= \hat{\mathbf{w}}^T (\mathbf{w}_{t-1} + y_t \phi(x_t)) \\ &\geq \hat{\mathbf{w}}^T \mathbf{w}_{t-1} + \gamma \\ &\geq \dots \\ &\geq \hat{\mathbf{w}}^T \mathbf{w}_0 + t\gamma \\ &\geq t\gamma\end{aligned}$$

ヨーシー・シュワルツの不等式 $(\|\hat{\mathbf{w}}\| \|\mathbf{w}_t\|)^2 \geq (\hat{\mathbf{w}}^T \mathbf{w}_t)^2$

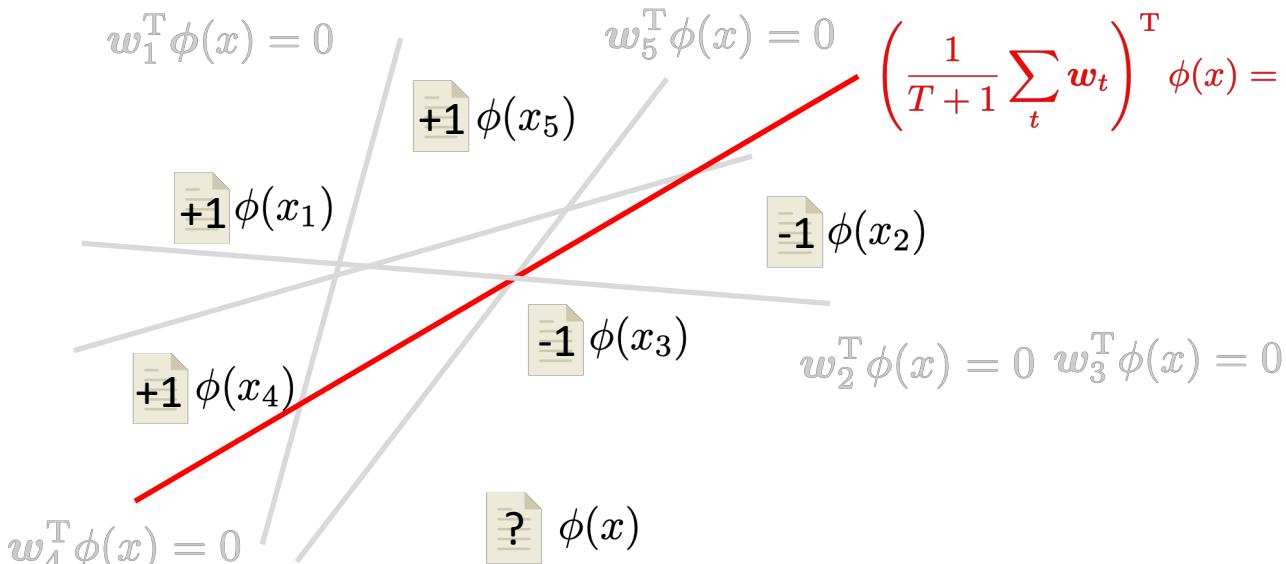
$$\|\mathbf{w}\|^2 tR^2 \geq t^2 \gamma^2$$

$$\frac{R^2}{\gamma^2} \geq t$$

平均化Perceptron [Freud+ 1998]

- Perceptron: 誤り駆動の線形分類器
 - 訓練例を誤分類したらパラメタ更新
 - 一定回数更新後終了
 - 線形分離可能なら有限回で収束
 - 学習結果は入力順に大きく依存
 - 重みベクトルの平均化により安定

```
w0 = 0
for t = 1 to T do
    if yt wt-1T φ(xt) ≤ 0
        wt = wt-1 + yt wt-1T φ(xt)
return 1/(T+1) ∑t wt
```



平均化Perceptron [Freud+ 1998]

- Perceptron: 誤り駆動の線形分類器
 - 訓練例を誤分類したらパラメタ更新
 - 一定回数更新後終了
 - 線形分離可能なら有限回で収束
 - 学習結果は入力順に大きく依存
 - 重みベクトルの平均化により安定
 - 空間効率よく計算可能

```
w0 = 0, w'0 = 0
for t = 1 to T do
    if ytwt-1Tϕ(xt) ≤ 0
        wt = wt-1 + ytwt-1Tϕ(xt)
        w't = w't-1 + tytwt-1Tϕ(xt)
return wT - 1/(T+1)w'T
```

$$\begin{aligned} \mathbf{w} - \frac{1}{T+1}\mathbf{w}'_T &= \mathbf{w}_T - \frac{1}{T+1} \sum_{t=1}^T t(\mathbf{w}_t - \mathbf{w}_{t-1}) \\ &= \mathbf{w}_T - \frac{1}{T+1}((\mathbf{w}_1 - \mathbf{w}_0) + 2(\mathbf{w}_2 - \mathbf{w}_1) \cdots + T(\mathbf{w}_T - \mathbf{w}_{T-1})) \\ &= \mathbf{w}_T - \frac{1}{T+1}(-\mathbf{w}_0 - \mathbf{w}_1 - \mathbf{w}_2 - \mathbf{w}_3 \cdots - \mathbf{w}_{T-1} + T\mathbf{w}_T) \\ &= \frac{1}{T+1} \sum_t \mathbf{w}_t \end{aligned}$$

Passive Aggressive (PA) アルゴリズム [Crammer+ 2006]

- PA: マージンに基づくオンライン学習器

- 訓練例を十分なマージンで正しく分類できなければパラメタ更新

- 訓練例ごとにヒンジ損失を最小化

$$L_{hinge}(\mathbf{x}_t) = \max\{0, 1 - y_t \mathbf{w}^T \phi(x_t)\} \rightarrow 0$$

$t = 2$

$$\mathbf{w}_1^T \phi(x) = 0$$

$$\begin{aligned} \mathbf{w}_t &= \operatorname{argmin}_{\mathbf{w}'} \frac{1}{2} \|\mathbf{w}' - \mathbf{w}_{t-1}\| \\ \text{s.t. } &1 - y_t \mathbf{w}'^T \phi(x_t) = 0 \end{aligned}$$

```
 $\mathbf{w}_0 = 0$ 
for  $t = 1$  to  $T$  do
    if  $y_t \mathbf{w}_{t-1}^T \phi(x_t) \leq 1$ 
         $\tau = \frac{1 - y_t \mathbf{w}_{t-1}^T \phi(x_t)}{\|\phi(x_t)\|^2}$ 
         $\mathbf{w}_t = \mathbf{w}_{t-1} + \tau y_t \mathbf{w}^T \phi(x_t)$ 
return  $\mathbf{w}_T$ 
```

+1 $\phi(x_1)$

-1 $\phi(x_2)$

$$\mathbf{w}_2^T \phi(x) = 0$$

パラメタ更新後は必ず
正しく分類できる

Passive Aggressive (PA) アルゴリズム [Crammer+ 2006]

- PA: マージンに基づくオンライン学習器

- 訓練例を十分なマージンで正しく分類できなければパラメタ更新

- 例ごとにヒンジ損失を最小化
- PA-I: 制約を緩めノイズに対処

$$\begin{aligned} \mathbf{w}_t = \operatorname{argmin}_{\mathbf{w}'} & \frac{1}{2} \|\mathbf{w}' - \mathbf{w}_{t-1}\|^2 + C\xi \\ \text{s.t.} & 1 - y_t \mathbf{w}'^T \phi(x_t) < \xi \quad (\xi > 0) \end{aligned}$$

$t = 2$

$$\mathbf{w}_1^T \phi(x) = 0$$

$$\begin{matrix} \pm 1 \\ \phi(x_1) \\ \phi(x_3) \end{matrix}$$

```
 $\mathbf{w}_0 = 0$ 
for  $t = 1$  to  $T$  do
    if  $y_t \mathbf{w}_{t-1}^T \phi(x_t) \leq 1$ 
         $\tau = \min \left\{ C, \frac{1 - y_t \mathbf{w}_{t-1}^T \phi(x_t)}{\|\phi(x_t)\|^2} \right\}$ 
         $\mathbf{w}_t = \mathbf{w}_{t-1} + \tau y_t \mathbf{w}_{t-1}^T \phi(x_t)$ 
return  $\mathbf{w}_T$ 
```

$$\begin{matrix} -1 \\ \phi(x_2) \end{matrix} \quad \mathbf{w}_2^T \phi(x) = 0$$

$$\mathbf{w}_3^T \phi(x) = 0$$

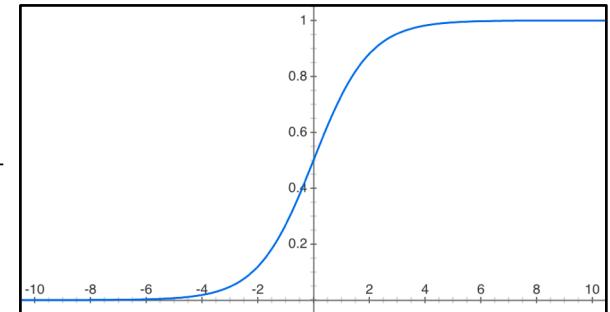
ロジスティック回帰

- ロジスティック回帰

線形分類器で sigmoid 関数 σ を使って確率を出力できるようにしたもの ($y = 1, 0$)

$$P(y|x) = \sigma(\phi(x)) \\ = \frac{1}{1 + e^{-z}}$$

$$y = \sigma(x) = \frac{1}{1 + e^{-x}}$$



$$z = \mathbf{w}^T \phi(x) = b + \sum_{i=1} w_i \phi(x)_i \quad (b = w_0, \forall x, \phi(x)_0 = 1)$$

- 歴史的経緯から様々な別名が存在するので注意

- 対数線形モデル (log-linear model)
- 最大エントロピー法 (Maximum Entropy, MaxEnt)

ロジスティック回帰の学習

- 誤分類のコストを損失関数で定義し, 学習データに対する損失を最小化するようパラメタを最適化
 - バッチ学習: 全訓練例に対し損失を計算
 - ミニバッチ学習: 訓練例の一部に対し損失を計算
 - オンライン学習: 訓練例ごとに損失を計算
PA はヒンジ損失を最小化するオンライン学習
- ロジスティック回帰の場合
 - 損失関数
クロスエントロピー損失
 - バッチ学習法
最急降下法 (gradient descent)
 - オンライン/ミニバッチ学習法
確率的最急降下法 (stochastic gradient descent)

クロスエントロピー損失

- 正解クラスの確率分布とモデルの推定するクラスの確率分布の間のクロスエントロピー
 - クロスエントロピー損失の最小化
= 学習データの条件付き対数尤度の最大化

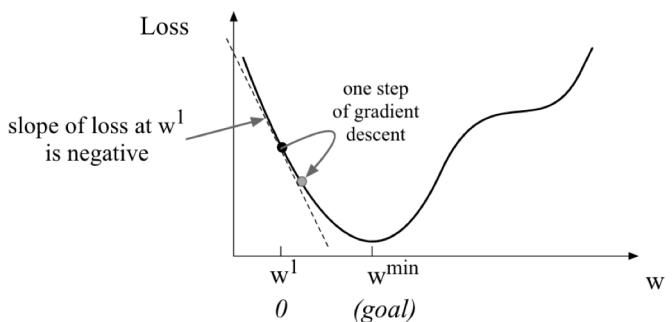
$$\begin{aligned}L_{\text{CE}}(\hat{y}, y' \mathbf{x}) &= -\log P(y|x) \\&= -\log \hat{y}^y (1 - \hat{y})^{1-y} \\&= -[y \log \hat{y} + (1 - y) \log(1 - \hat{y})] \\&= -[y \log \sigma(\mathbf{w}^T \phi(x)) + (1 - y) \log(1 - \sigma(\mathbf{w}^T \phi(x)))]\end{aligned}$$

勾配降下法 (Gradient Descent)

- 全学習データに対する損失の微分から損失関数の勾配を計算し、パラメタを逐次更新

$$\hat{\mathbf{w}} = \operatorname{argmin}_{\mathbf{w}} \frac{1}{m} \sum_{i=0}^m L_{\text{CE}}(\hat{y}_i, y_i; \mathbf{w})$$

$$\mathbf{w}_t = \mathbf{w}_{t-1} - \eta \frac{\partial}{\partial \mathbf{w}} \left(\frac{1}{m} \sum_{i=0}^m L_{\text{CE}}(\hat{y}_i, y_i; \mathbf{w}) \right)$$



$$\frac{\partial}{\partial w_j} L_{\text{CE}}(\hat{y}_i, y_i; \mathbf{w}) = [\sigma(\mathbf{w}^T \phi(x_i)) - y_i] x_j$$

ロジスティク関数では損失関数が凸関数のため大域最適解が求まる

勾配降下法の勾配の導出

$$\begin{aligned}\frac{\partial}{\partial w_j} L_{\text{CE}}(\hat{y}, y; \mathbf{w}) &= \frac{\partial}{\partial w_j} - [y \log \sigma(\mathbf{w}^T \phi(x)) + (1-y)(1-\sigma(\mathbf{w}^T \phi(x)))] \\ &= - \left[\frac{\partial}{\partial w_j} y \log \sigma(\mathbf{w}^T \phi(x)) + \frac{\partial}{\partial w_j} (1-y)(1-\sigma(\mathbf{w}^T \phi(x))) \right]\end{aligned}$$

$$\begin{aligned}\frac{d \ln(x)}{d} = \frac{1}{x} &= -\frac{y}{\sigma(\mathbf{w}^T \phi(x))} \frac{\partial}{\partial w_j} \sigma(\mathbf{w}^T \phi(x)) - \frac{1-y}{1-\sigma(\mathbf{w}^T \phi(x))} \frac{\partial}{\partial w_j} 1 - \sigma(\mathbf{w}^T \phi(x)) \\ &= - \left[\frac{y}{\sigma(\mathbf{w}^T \phi(x))} - \frac{1-y}{1-\sigma(\mathbf{w}^T \phi(x))} \right] \frac{\partial}{\partial w_j} \sigma(\mathbf{w}^T \phi(x))\end{aligned}$$

$$\begin{aligned}\frac{d\sigma(z)}{dz} = \sigma(z)(1-\sigma(z)) &= - \left[\frac{y - \sigma(\mathbf{w}^T \phi(x))}{\sigma(\mathbf{w}^T \phi(x))(1-\sigma(\mathbf{w}^T \phi(x)))} \right] \sigma(\mathbf{w}^T \phi(x))(1-\sigma(\mathbf{w}^T \phi(x))) \frac{\partial(\mathbf{w}^T \phi(x))}{\partial w_j} \\ &= - \left[\frac{y - \sigma(\mathbf{w}^T \phi(x))}{\sigma(\mathbf{w}^T \phi(x))(1-\sigma(\mathbf{w}^T \phi(x)))} \right] \sigma(\mathbf{w}^T \phi(x))(1-\sigma(\mathbf{w}^T \phi(x))) x_j \\ &= -[y - \sigma(\mathbf{w}^T \phi(x))] x_j \\ &= [\sigma(\mathbf{w}^T \phi(x)) - y] x_j\end{aligned}$$

確率的勾配降下法 (stochastic gradient descent)

- 一部学習データに対する損失の微分から損失関数の勾配を近似計算し、重み更新
 - 更新幅は学習率 η_t で調整 (時間減衰させるのが一般的)
 - ミニバッチ化する場合は複数の訓練例を使い損失を計算

```
w0 = 0
for t = 1 to T do
    estimate  $\hat{y}_t = f(\mathbf{x}_t; \mathbf{w}_{t-1})$ 
     $\mathbf{w}_t = \mathbf{w}_{t-1} - \eta_t \frac{dL(\hat{y}_t, y_t)}{d\mathbf{w}}$ 
return  $\mathbf{w}_t$ 
```

過学習と正則化

- 過学習 (overfitting)
モデルが学習データに強く適合し、未知のテストデータに対する汎化性能が下がる
- 正則化 (regularization)
極端なパラメタの値を抑制する罰則項(正則化項)を損失関数に追加

$$\hat{\boldsymbol{\theta}} = \operatorname{argmin}_{\boldsymbol{\theta}} \frac{1}{m} \sum_{i=1}^m L_{\text{CE}}(\hat{y}_i, y_i; \boldsymbol{\theta}) - \alpha R(\boldsymbol{\theta})$$

ハイパーパラメタ

$$\begin{cases} \|\boldsymbol{w}\|_2^2 = \sum_j w_j^2 & L2 \text{ 正則化 (ridge 回帰)} \\ \|\boldsymbol{w}\|_1 = \sum_j |w_j| & L1 \text{ 正則化 (lasso 回帰)} \end{cases}$$

- ベイズ的な解釈も可能(パラメタの事前分布を仮定し、事後分布を最大化; L2: ガウス分布, L1: ラプラス分布)

多クラスロジスティック回帰

- 多クラスロジスティック回帰

ロジスティック回帰で sigmoid 関数の代わりに softmax 関数を用いることで多クラス化したもの

$$P(y = c|x) = \frac{e^{z_c}}{\sum_{c' \in C} e^{z_{c'}}}$$
$$z_c = b_c + \mathbf{w}_c^T \boldsymbol{\phi}(x)$$

- 素性関数: 訓練例のクラスに応じて発火(素性数|C|倍)
- 最適化: 2クラスの場合とほぼ同様

分類問題の評価尺度 (1/2)

- 特定クラスの分類評価: Precision/Recall/F-measure
 - 分類正解: 真陽性 (true negative), 真陰性 (true negative)
 - 分類誤り: 偽陽性 (false positive), 偽陰性 (false negative)
- 全体の評価: accuracy, 上記のマクロ/マイクロ平均
confusion matrix: 分類結果を正解/出力ラベルに従い分類正解 (gold label)

	肯定的	否定的	
分類器の出力	肯定的	true positive (tp)	false positive (fp)
	否定的	false negative (fn)	true negative (tn)
	$\text{Recall} = \frac{tp}{tp + fn}$		$\text{Precision} = \frac{tp}{tp + fp}$
			$\text{Accuracy} = \frac{tp + tn}{tp + fp + tn + fn}$

分類問題の評価尺度 (1/2)

- 特定クラスの分類評価: Precision/Recall/F-measure
 - 分類正解: 真陽性 (true negative), 真陰性 (true negative)
 - 分類誤り: 偽陽性 (false positive), 偽陰性 (false negative)
- 全体の評価: accuracy, 上記のマクロ/マイクロ平均
confusion matrix: 分類結果を正解/出力ラベルに従い分類正解 (gold label)

	肯定的	否定的	
分類器の出力	肯定的	$F_\beta = \frac{1}{\alpha \frac{1}{P} + (1 - \alpha) \frac{1}{R}}$ $= \frac{(\beta^2 + 1)PR}{\beta^2 P + R}$ $\beta = \frac{1 - \alpha}{\alpha}$	$Precision = \frac{tp}{tp + fp}$
	否定的	$Recall = \frac{tp}{tp + fn}$	$Accuracy = \frac{tp}{tp + fp + tn + fn}$

分類問題の評価尺度 (2/2)

- 特定クラスの分類評価: Precision/Recall/F-measure
 - 分類正解: 真陽性 (true negative), 真陰性 (true negative)
 - 分類誤り: 偽陽性 (false positive), 偽陰性 (false negative)
- 全体の評価: accuracy, 上記のマクロ/マイクロ平均
例) (中立ラベルありの)評判分析

正解 (gold labels)

分類器の出力	肯定的	中立	否定的		
	肯定的	128	16	8	152
	中立的	26	74	7	107
	否定的	6	20	115	141
	160	110	130	400	

肯定ラベル

$$\text{Precision: } 128/152 = 0.842$$

$$\text{Recall: } 128/160 = 0.800$$

$$F_1 = 2 \cdot P \cdot R / (P + R) = 0.809$$

$$\text{Accuracy: } (128+74+115)/400 = 0.8175$$

交差検定 (cross-validation)

- 交差検定: 学習・開発/テストデータを入れ替えて全データを評価に使う頑健な評価方法
 - 教師あり学習では学習データを大きくしたい
 - 評価の信頼性を高めるためテストデータも大きくしたい

5-fold cross validation

test	train		
t	test	rain	
tr		test	ain
tra		test	in
train			test

仮説検定

- 手法A,B間の性能差に統計的な有意差があるか？
- 仮説検定: 手法間の性能差が偶然発生(帰無仮説)したとする確率 (*p-value*) を適切な有意水準の元に棄却
 - 棄却する *p-value* の閾値は経験的に 0.05 or 0.01
- 統計的自然言語処理で用いるべき検定手法 [Dror+ 2018]
 - 評価指標が何らかの確率分布に従う: paired student's t-test
 - 評価指標が確率分布に従わない: ノンパラメトリック
 - 評価データ小: 並べ替え検定, ブートストラップ検定
 - 評価データ大: Wilcoxon 符号順位検定, 符号検定 (McNemar 検定)

仮説検定がない, あるいは誤った検定を行っている場合も

ブートストラップ検定

[Efron+ 1993, Berg-Kirkpatrick+ 2012]

- 任意のタスク・評価尺度で手法 A と手法 B の性能の有意差を検定する手法

- テストデータ x から重複を許し要素を n 回に乱択して
ブートストラップサンプル $x^{*(i)}$ を b 個得る
- $x^{*(i)}$ での性能差が $\delta(x^{*i}) > 2\delta(x)$ となる回数 s を計算
- $p\text{-value} \approx \frac{s}{b}$ と計算

例) サイズ10のテストセットについて分類精度を評価する場合

	1	2	3	4	5	6	7	8	9	10	A%	B%	$\delta()$
x	AB	.7	.5	.2									
$x^{*(1)}$	AB	.6	.6	.0									
$x^{*(1)}$	AB	.6	.7	-.1									
...													
$x^{*(b)}$													

Bias と Variance

- 分類器の性能は素性関数と学習データサイズに依存
 - 素性数少: モデルの近似が強い (Bias 大)
 - データ少: パラメタの推定値の分散大 (Variance 大)
Bias と Varianceはトレード・オフの関係
- Bias を減らすには
 - 素性関数を複雑化する (組合せ素性, 非線形関数)
- Variance を減らすには
 - クラウドソーシングの利用
 - 大量に利用できる自然注釈(部分正解)などの活用

本日のまとめ

- 言語モデル: 言語の自然さのモデル化
 - n-gram モデル
 - スムージングとバックオフ
 - 評価尺度: test-set perplexity
- 分類: 言語をカテゴリ(クラス)に分ける
 - 生成モデル: Naïve Bayes, Complement Naïve Bayes
 - 識別モデル: Perceptron, Passive Aggressive, ロジスティック回帰
 - 評価尺度: Accuracy, Precision/Recall/F-measure
- 交差検定・仮説検定