

機械学習モデルによる ツイート分類

遠藤 皓義



課題

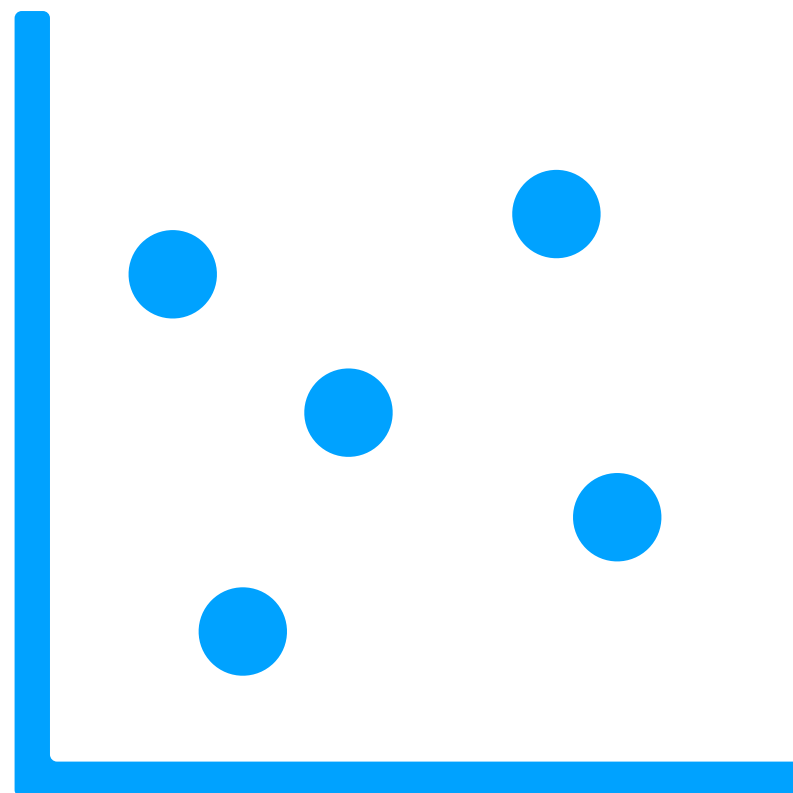
ツイッターのツイートを
以下のクラスに分類するモデル作成

hate speech

offensive language

neither

データ紹介



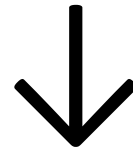
データ紹介



Sharkevin RIP Sean P
@WheresMyJuice

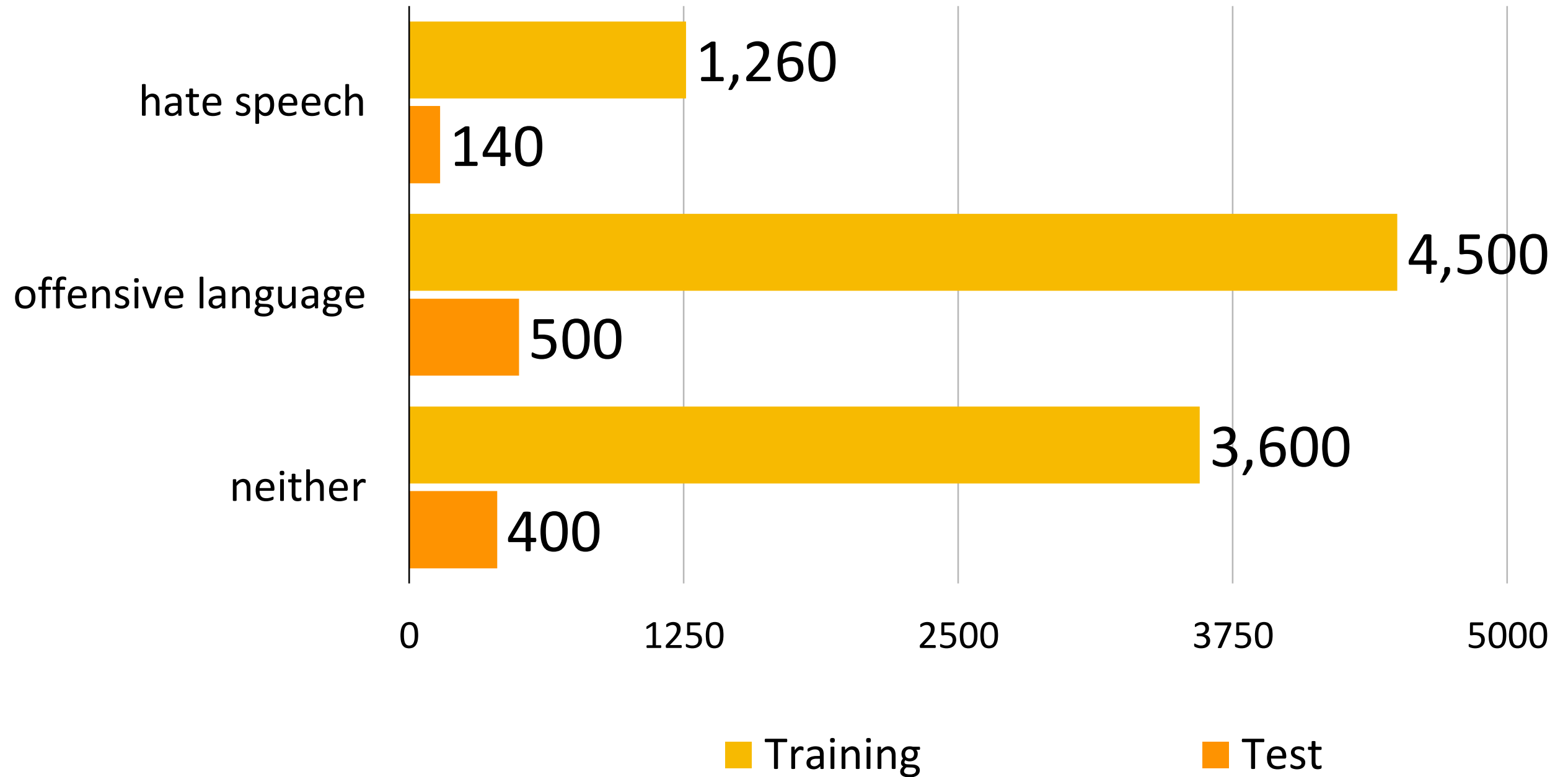
cookies and crackers aren't even on the same level!

午前7:04 · 2014年6月27日 · Twitter Web Client



cookies and crackers aren't even on the same level!

クラス別データ数

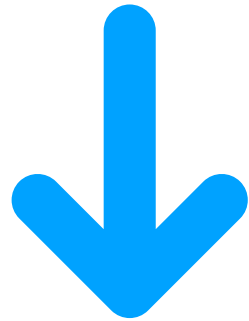


Google翻訳による
データ水増し



原文：

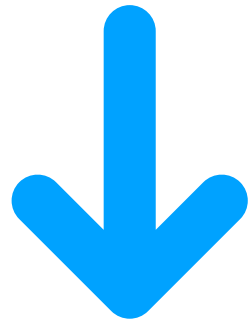
Fuck you you pussy ass hater go suck a dick and die fast



Google 翻訳でフランス語、ドイツ語、
オランダ語に翻訳

フランス語翻訳後：

Va te faire foutre le cul chatte aller sucer une bite et mourir vite



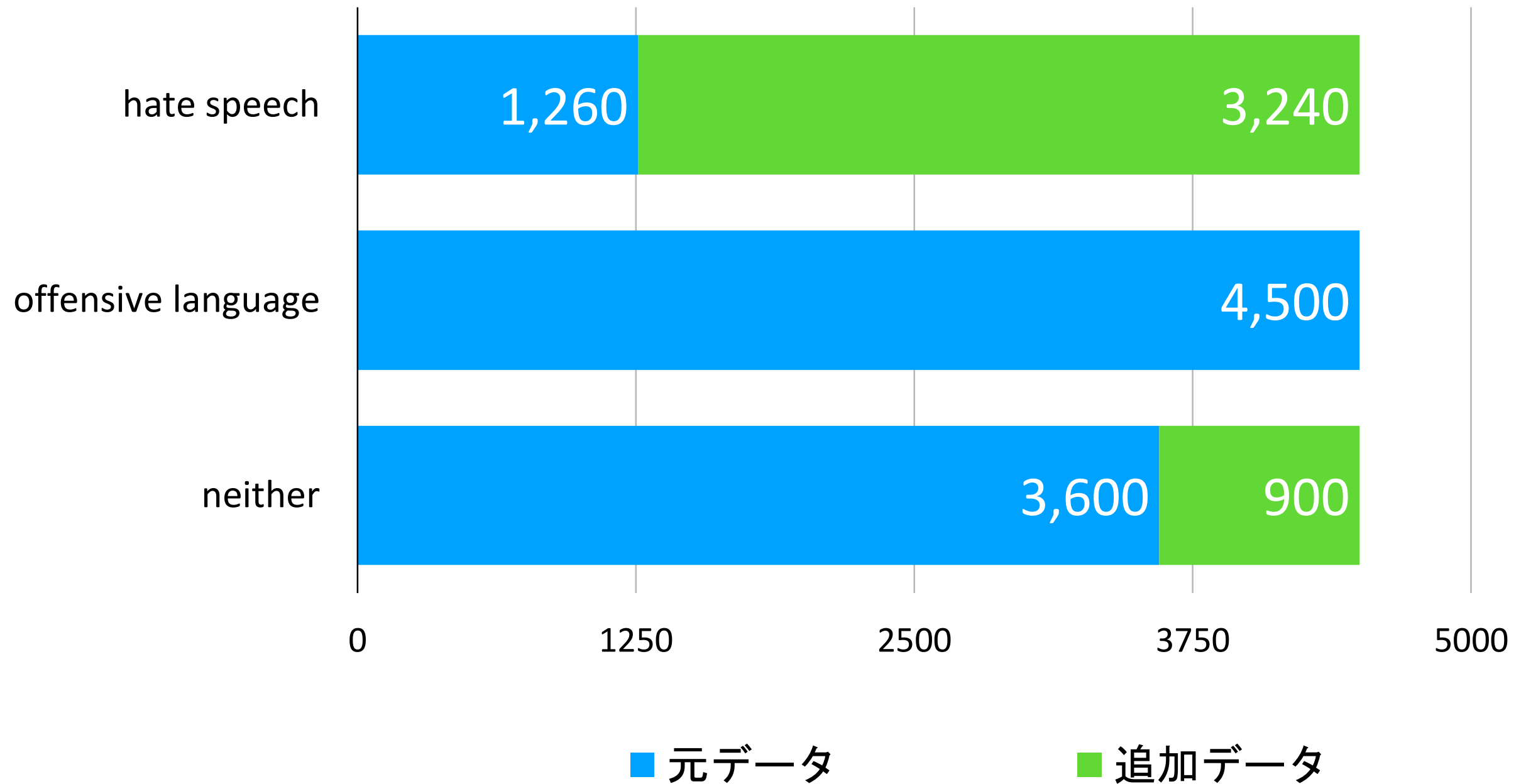
再び英語に翻訳

再翻訳後：

Fuck you pussy hater will suck a dick and die quick

データ増し後

8



テキスト前処理:置き換え⁹

"@John: How was Japan?" Awesome ! 😀
RT@Hana miss Japan :(😊)

@メンションを<user>

返信を示す

"<user>: How was Japan?" <reply> Awesome! :grinning_face:
<rt_from> miss Japan :sad_face:

リツイートを示す

絵文字を内容で置き換え

テキスト前処理 : lemmatization

- Stanford CoreNLP Package¹ を使用

→ 言語データの処理を一括で行える
Pipeline

- GATE Twitter POSTagger²モデル
によりPOS タグ付け

→ twitterのツイートで学習を行った
POS tagging モデル

(packageに別途追加)

- それを元にlemmatization

He ate cakes
↓ ↓ ↓
PRP VBD NNS
↓
He eat cake

1. <https://stanfordnlp.github.io/CoreNLP/extensions.html>

2. <https://gate.ac.uk/wiki/twitter-postagger.html>

データ前処理:その他

- スペルミス直し

→ モジュール: autocorrect¹ を使用
内蔵単語リストの中で最もレーベンシュタイン距離の近い物を返す（距離2以内）

- 短縮形を戻す

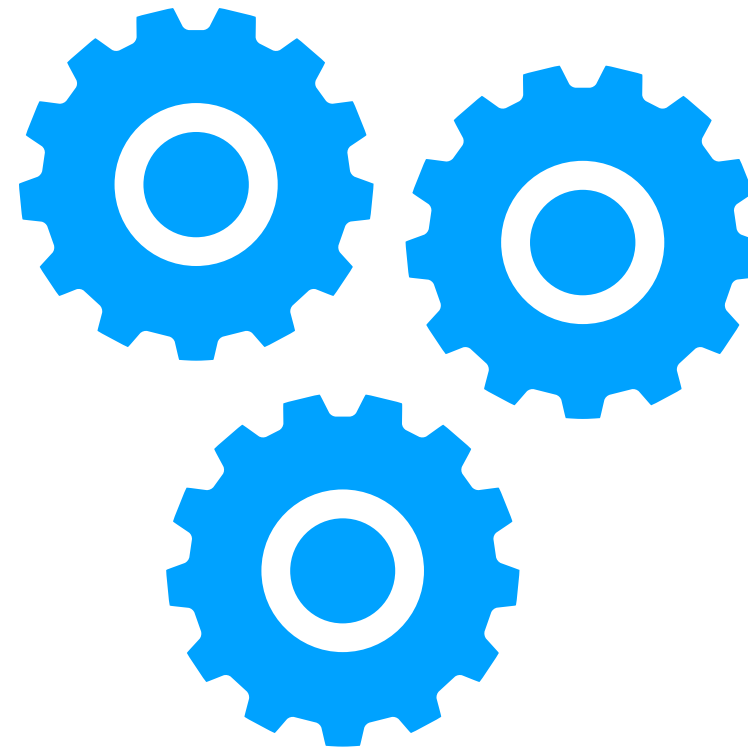
→ モジュール: contractions² を使用
単純な辞書型置き換え

- ?!以外の句読点削除

- 小文字に変換

- Stopwords削除

→ nltk³ 付属のリストを使用



1. <https://pypi.org/project/autocorrect/>
2. <https://pypi.org/project/contractions/>
3. <https://www.nltk.org/>

用いたモデル

- 決定木
- LSTM



決定木パラメーター

	元データのみ	水増しあり
最大深度	20	30
分岐に必要な最低 サンプル数	17.5%	15.5%
各分岐で考慮する 特徴量	84%	82%

LSTM

パラメーター・モデル構成

	元データのみ	水増し後
embedding	20	20
LSTM Blocks	91	171
dropout	47%	63%
Softmax	3	3
学習率	0.001	0.001

モデル構成

1. embedding 層

2. LSTM 層

オプティマイザー: Adam

活性化関数: layer \rightarrow tanh

gates \rightarrow hard sigmoid

3. Dropout

4. 出力層 : softmax

特徴抽出

- 決定木：TF-IDF
 - A. 全文書を通してレアな単語に高い重み
 - B. 文書出現頻度上位30%、下位0.4%の単語は排除
 - C. ユニグラムとバイグラムの組み合わせを使用

特徴抽出

- LSTM : モデルに単語の分散を学習させる
→タスクに適した分散を設定させる事が狙い

ex. 'white' と最もcosine 距離の近い単語

GloVe¹ (ツイッターベース) : black, blue, green, yellow, red, purple, tank

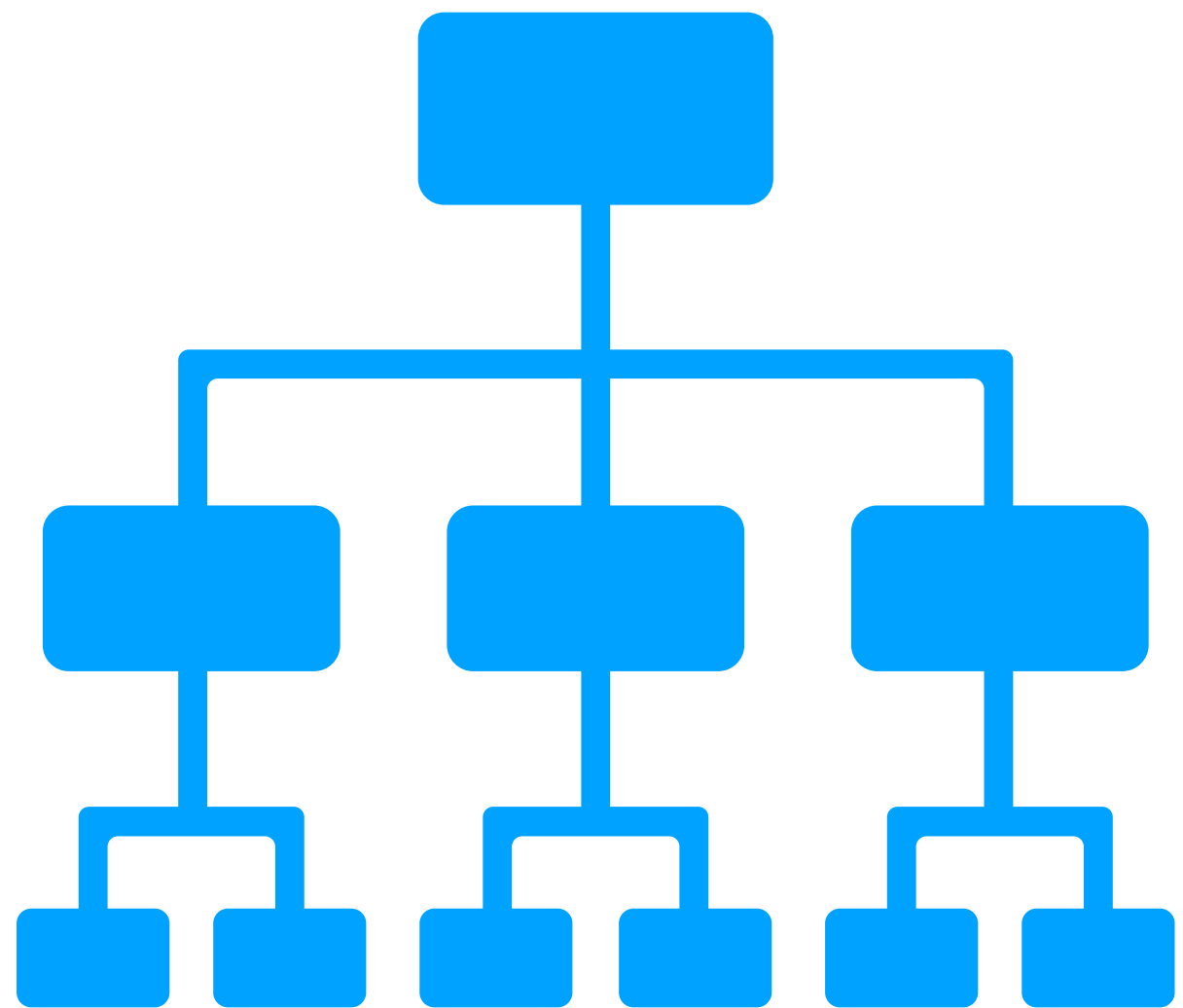
モデルが学習した分散 : nigga, fuck, faggot, nigger, fag, ass, racist

1. <https://nlp.stanford.edu/projects/glove/>

検証結果



決定木 CART



結果：元データのみ

19

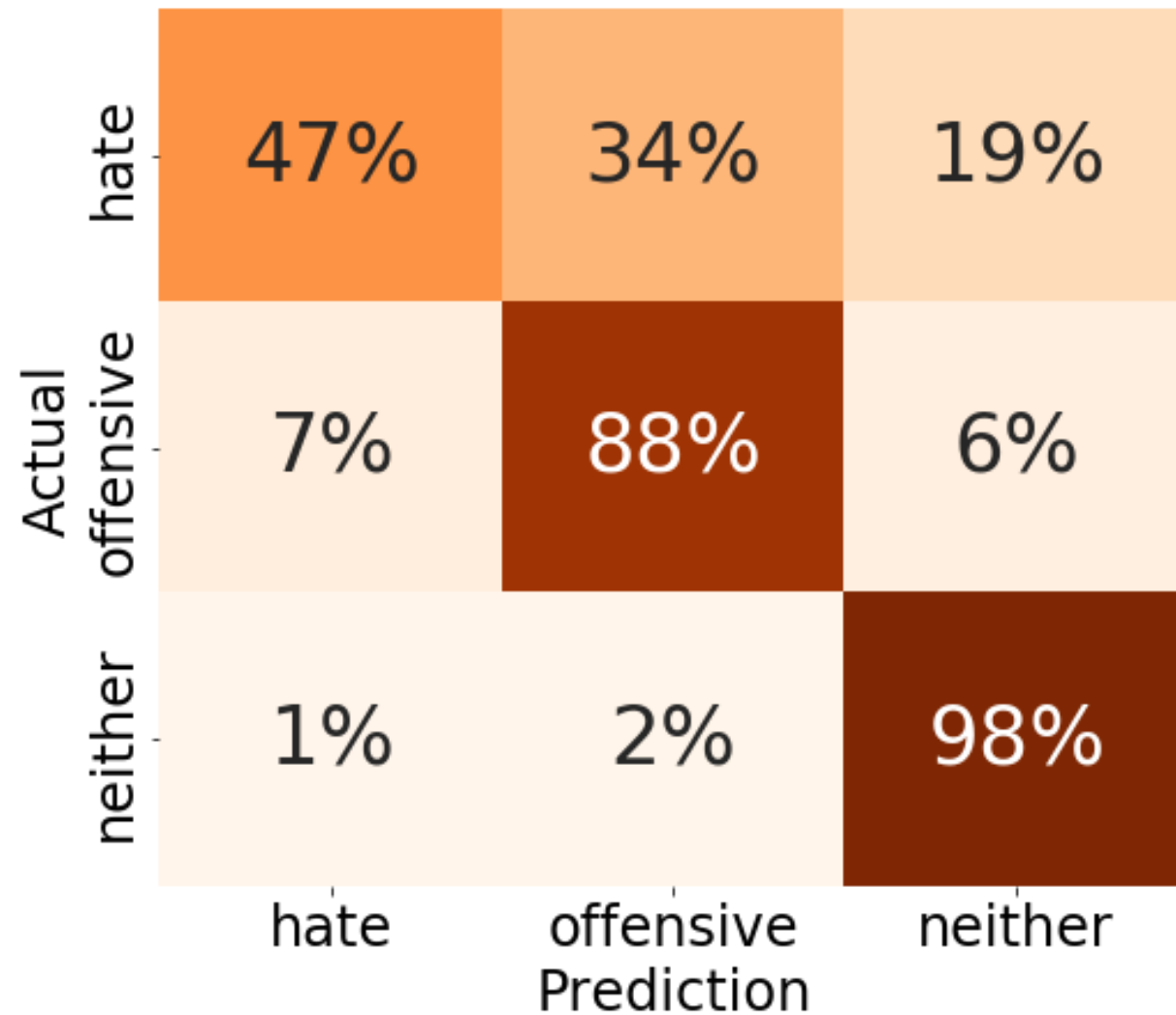
テストデータ分類結果 (%)

Actual	hate	offensive	neither	
	47%	34%	19%	
	7%	88%	6%	
neither	1%	2%	98%	
		hate	offensive	neither
		Prediction		

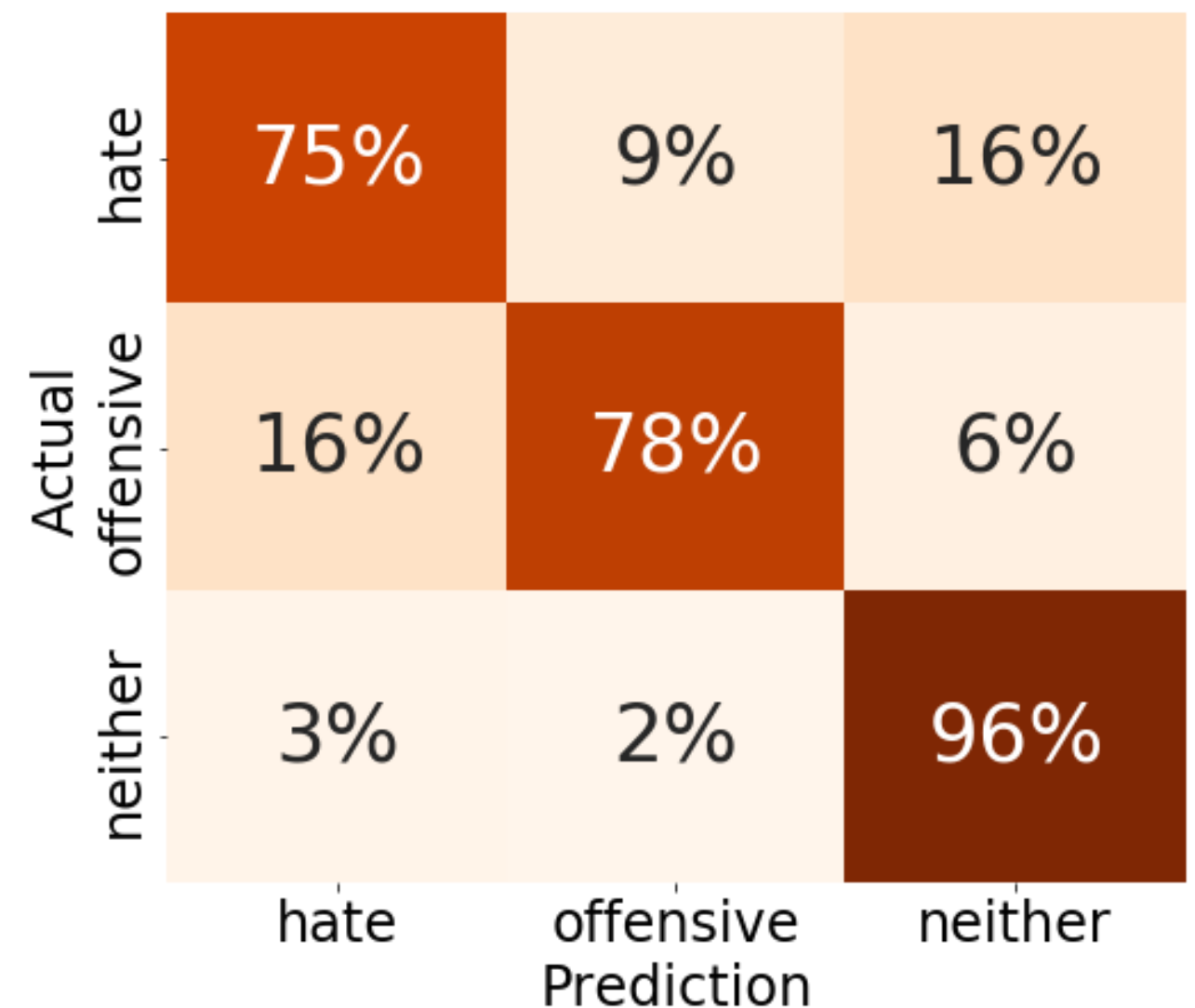
- offensive language, neitherは良く検出できている
f1-score 約90%
- hate speech の検出が 上手くいっていない
- 特に hate vs offensive に課題
- f1-macro: 78%
- f1-weighted: 85%

結果：水増し後

水増し前



水増し後



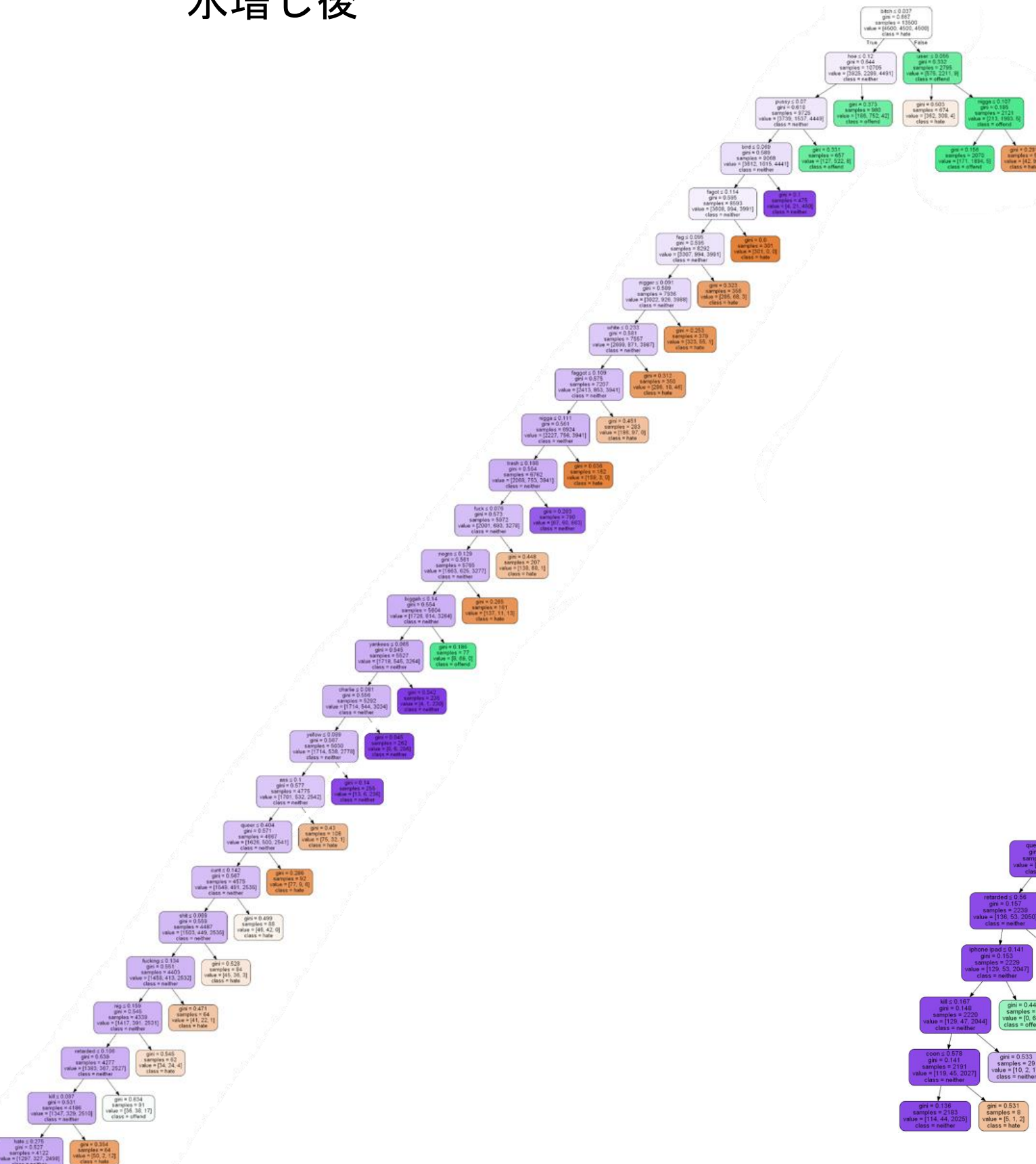
- hate speech の検出率が大幅に上昇

- 他のクラスの手増し speech への誤分類が増加

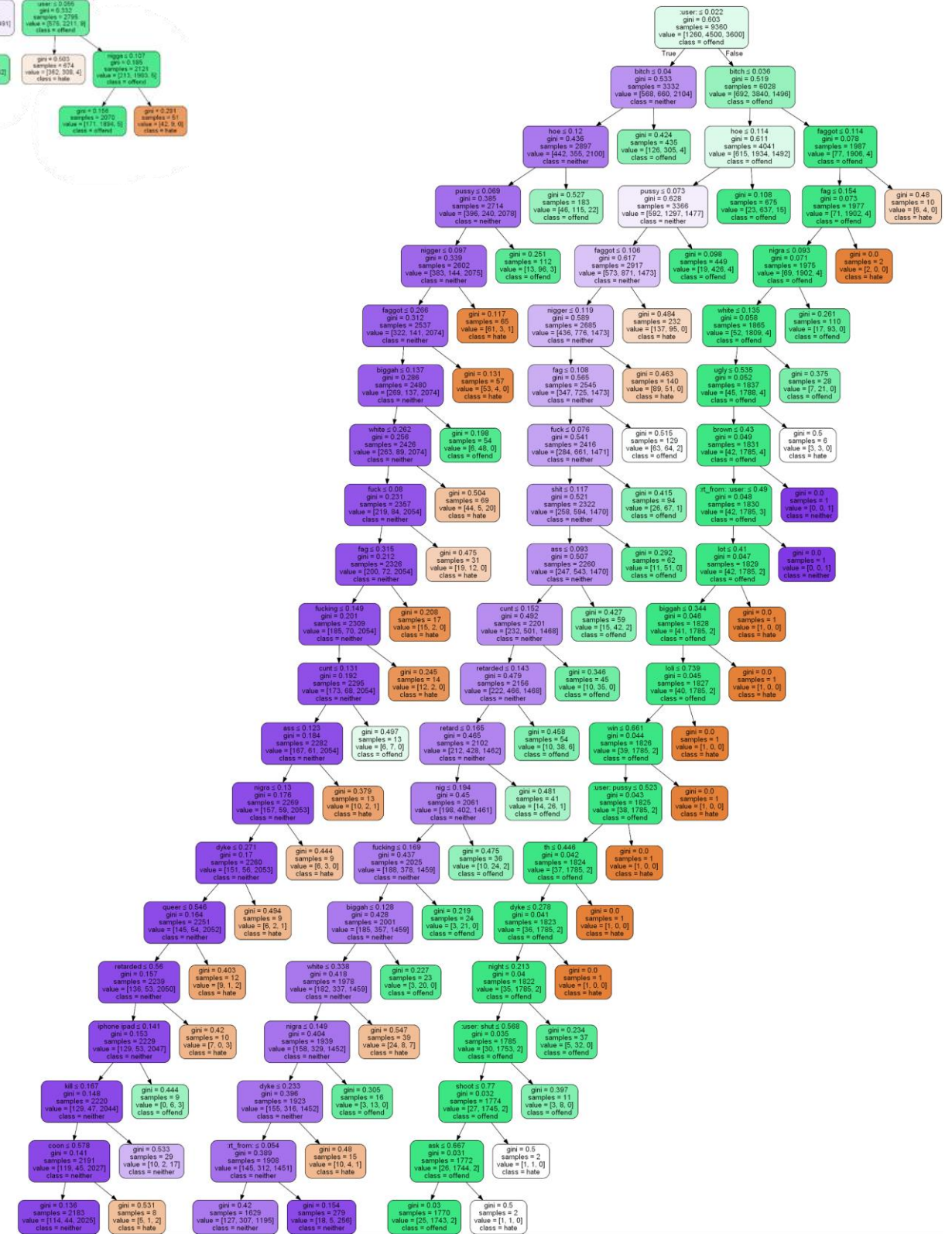
- f1-macro: 78% → 80%

- f1-weighted: 85% → 85%

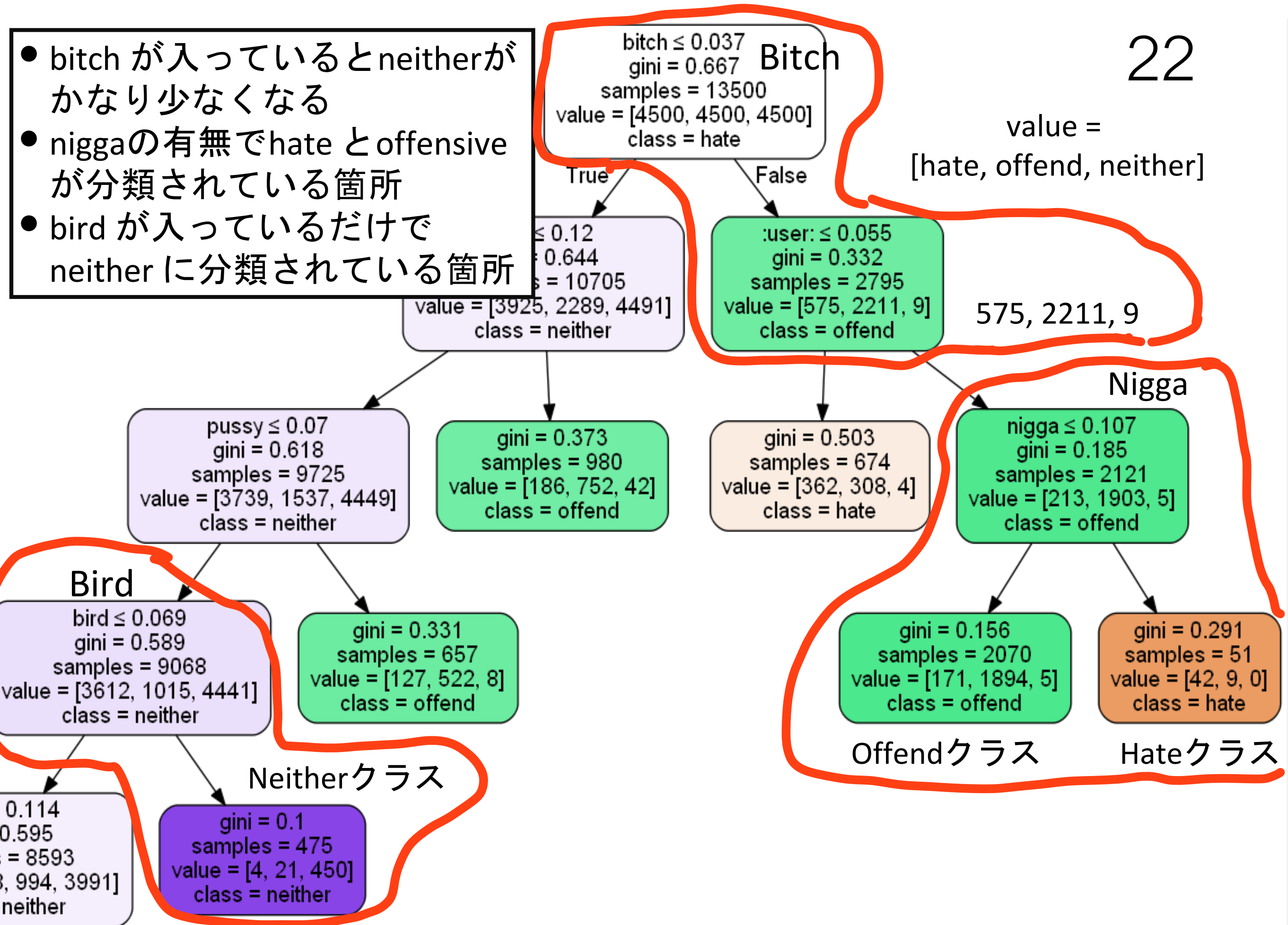
水増し後



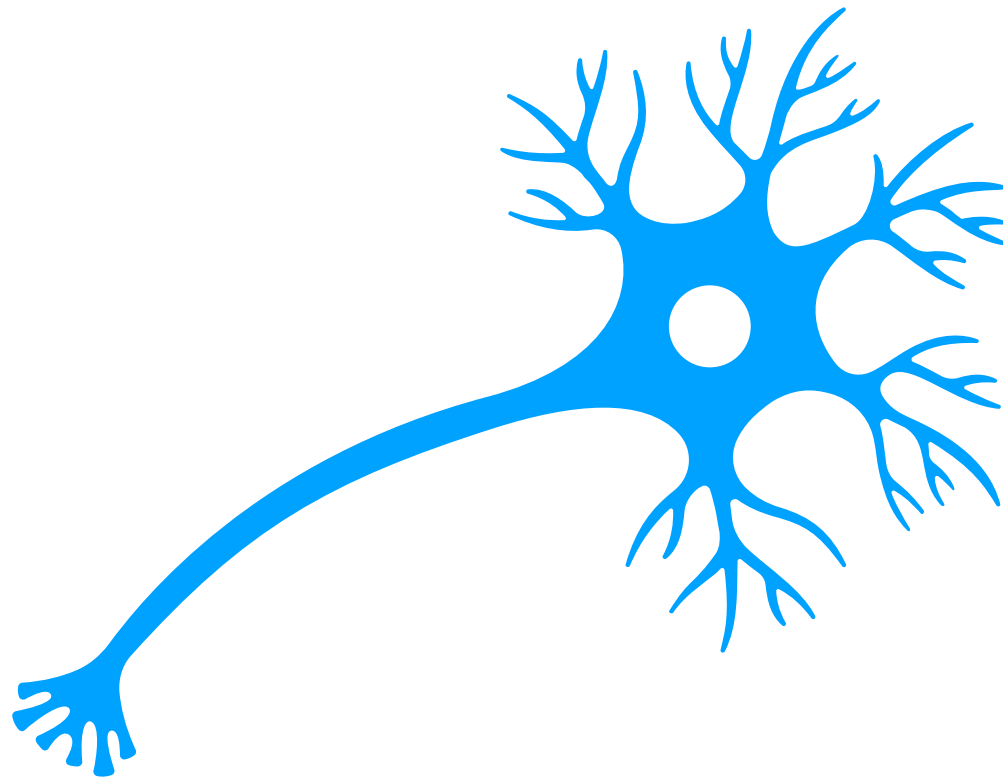
水増し前



- bitch が入っているとneitherがかなり少なくなる
- niggaの有無でhate とoffensiveが分類されている箇所
- birdが入っているだけでneitherに分類されている箇所



LSTM



結果: 元データのみ

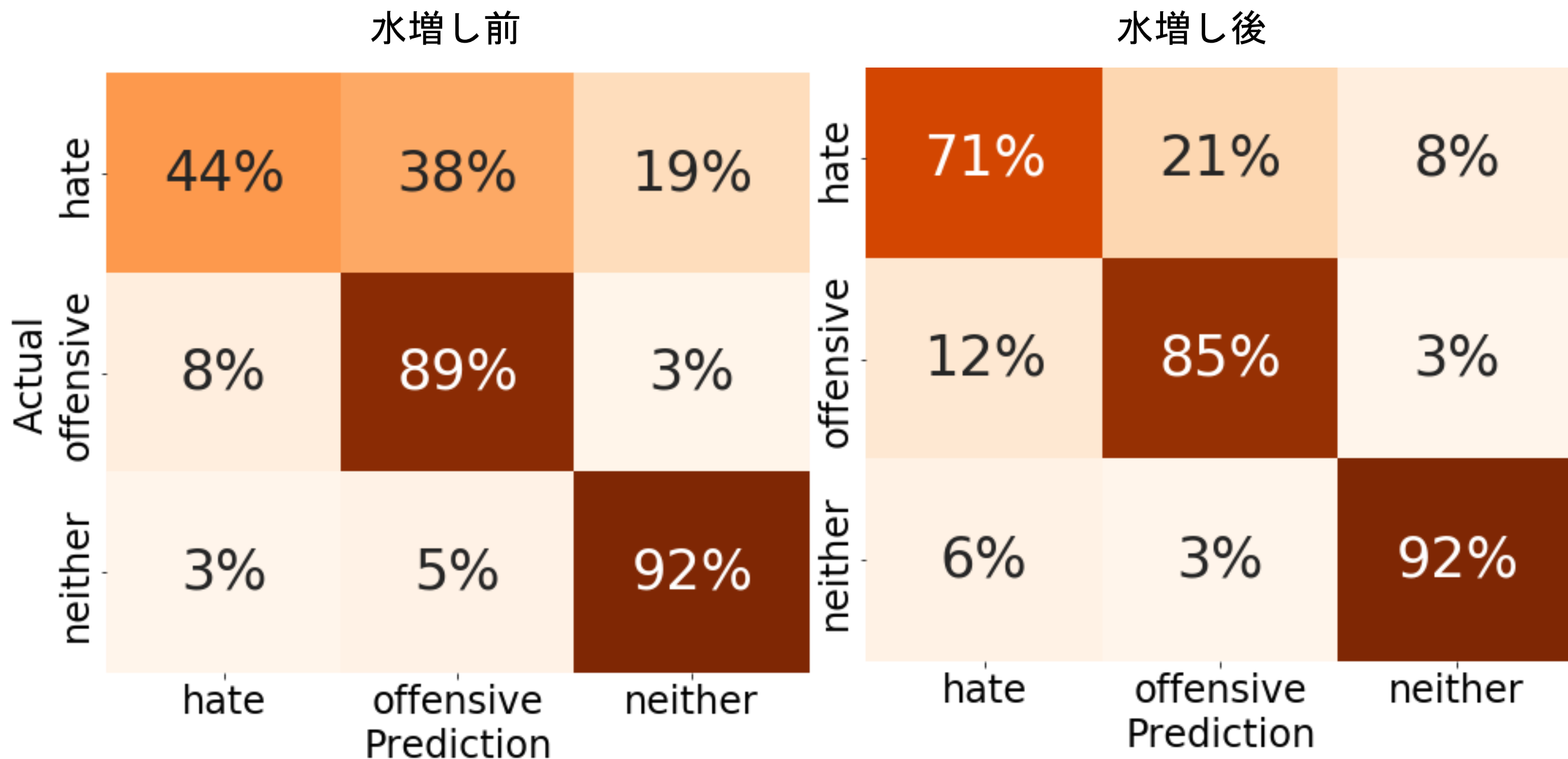
テストデータ分類結果 (%)

Actual	hate	offensive	neither
	44%	38%	19%
	8%	89%	3%
neither	3%	5%	92%
Prediction			

- 決定木と同じく offensive, neither クラスはよく分類されている
f1-score 約90%
- hate speech クラスはやはり上手く検出できていない
→ 38%がoffensive と誤分類
- f1-macro: 76%
- f1-weighted: 83%

結果: 水増し後

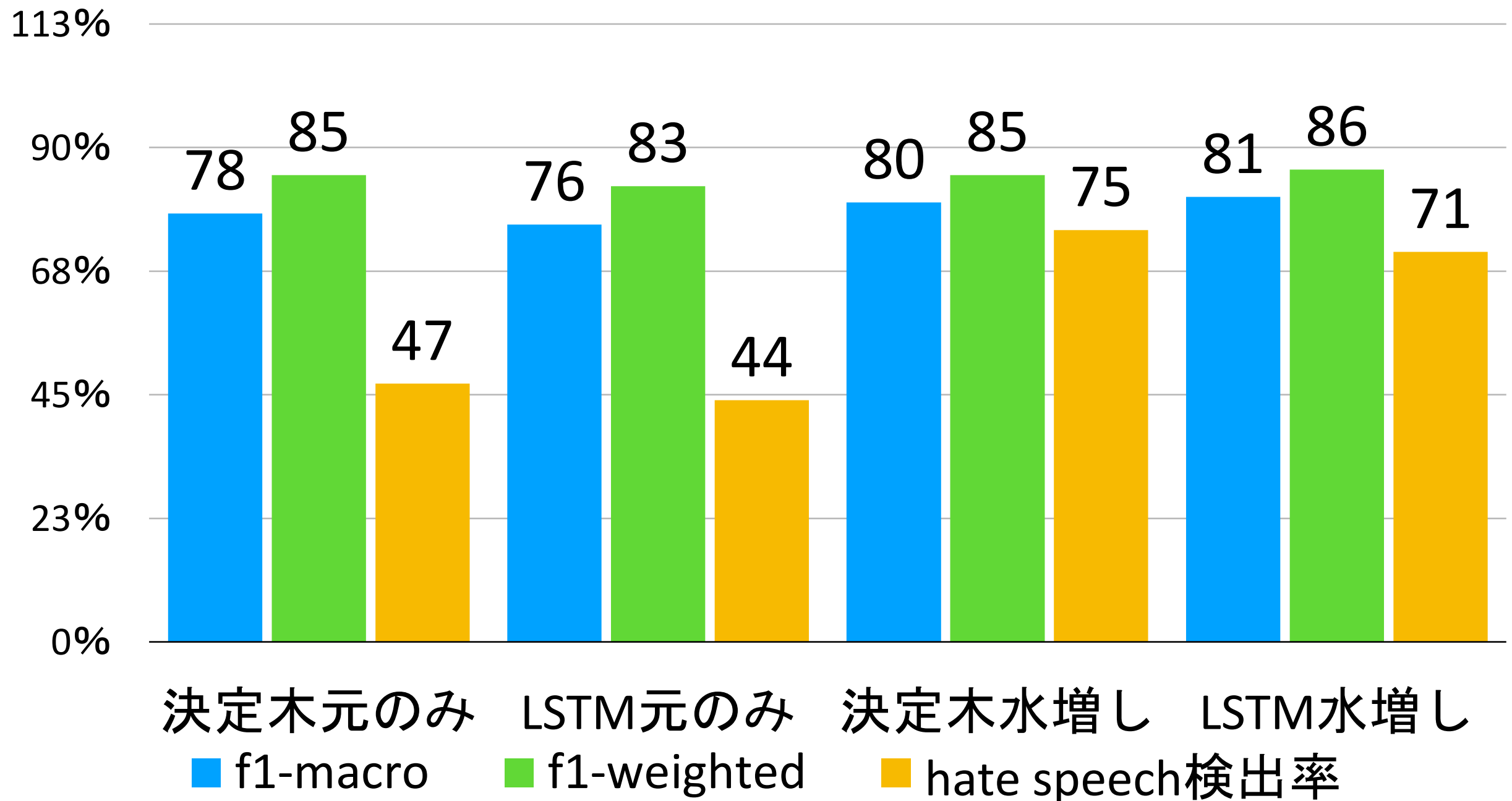
25



- hate speechクラスの検出率が向上
- hate speech の誤分類量が増加

- f1-macro: 76% → 81%
- f1-weighted: 83% → 86%

モデル比較



- 決定木、LSTM共に同等の精度

結論



まとめ

- 決定木、LSTMを用いて分類
- offensive language 及び neither クラス は上手く検出できる
 - それのみを目的とすれば2つとも非常によい
- hate speech の区別には課題

結論: hate vs offensive

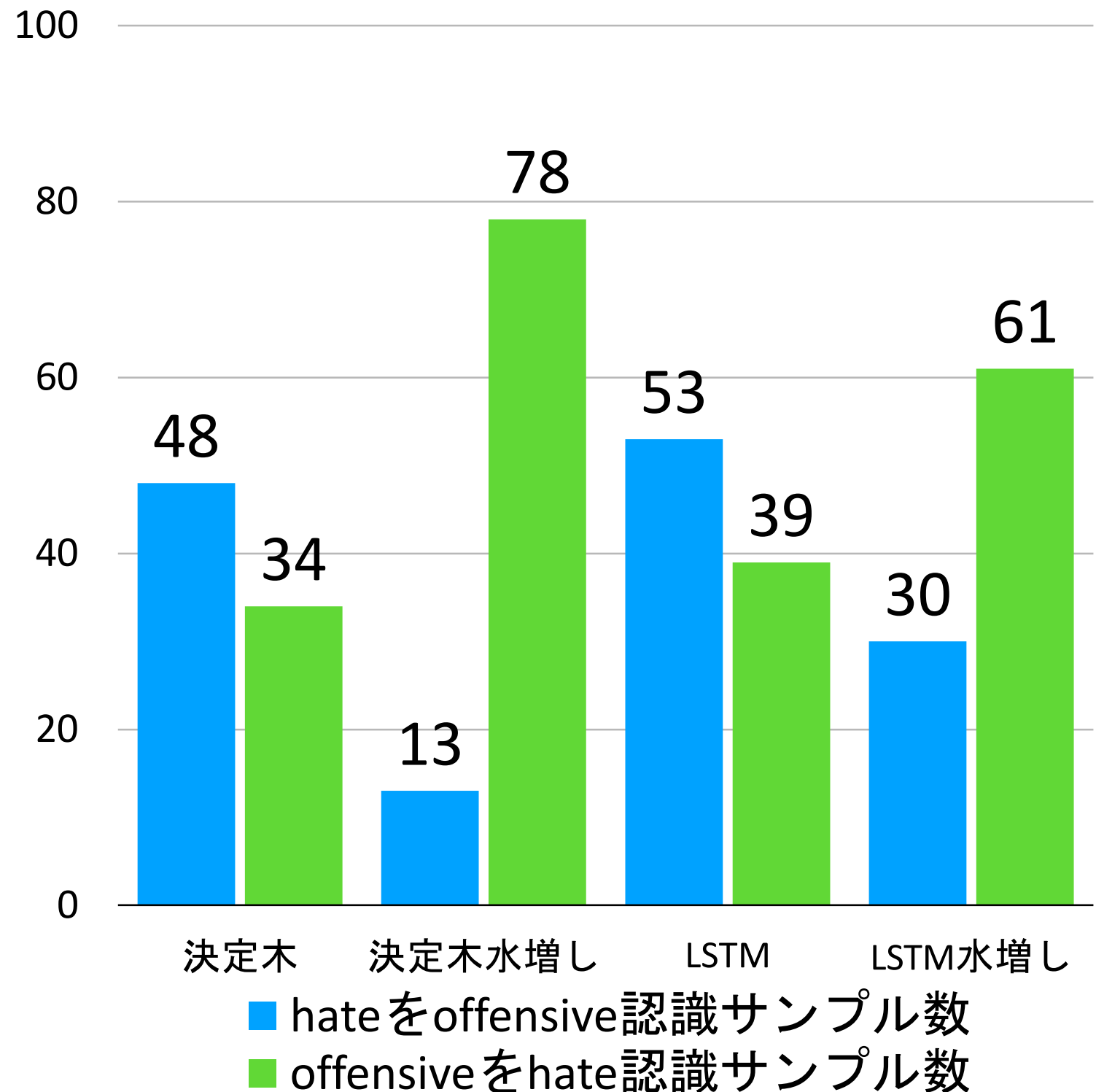
- この二つのクラスの区別がモデルには難しい

- 元データのみ

→hate speech がoffensive language として分類されやすい

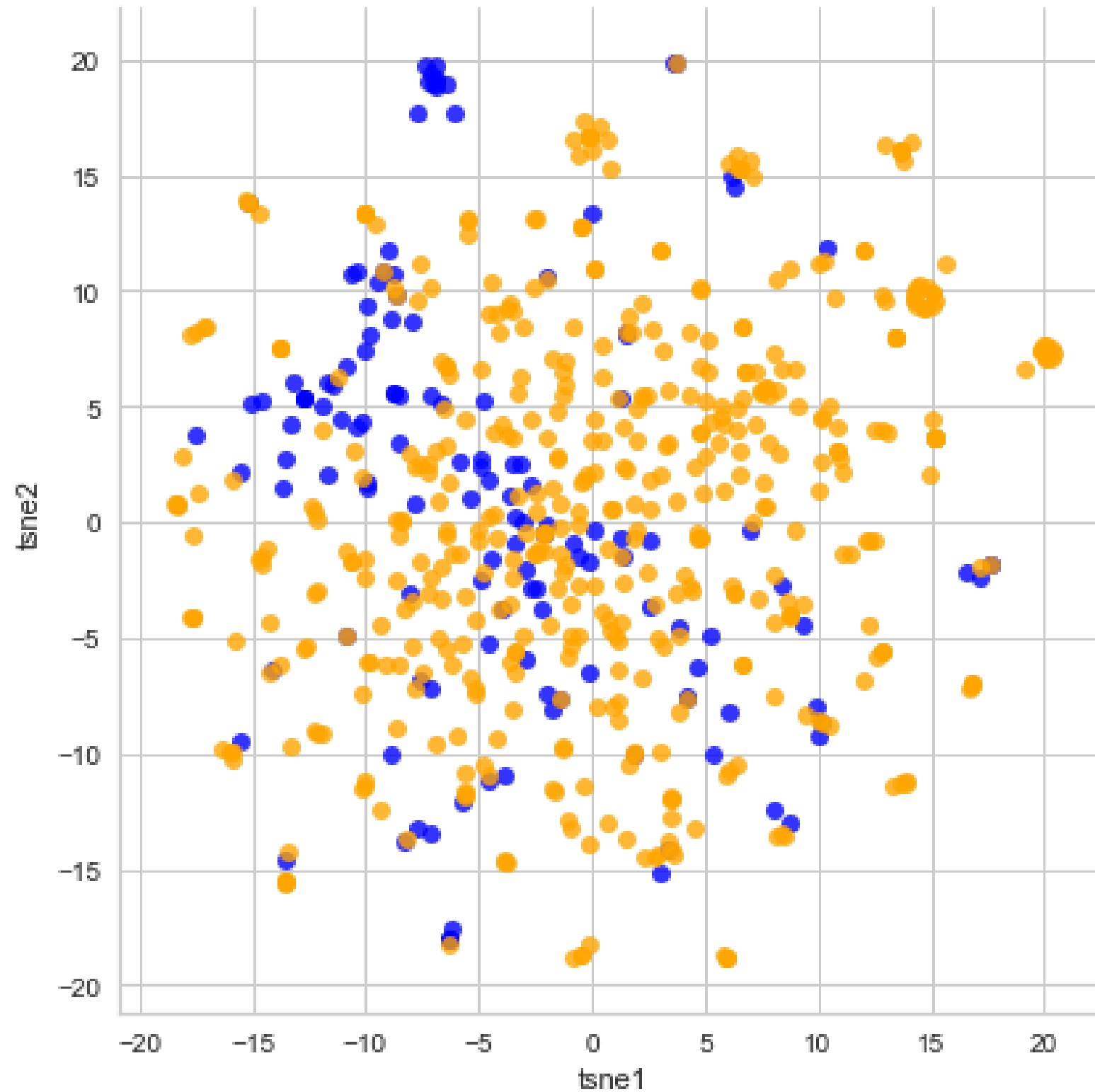
- クラスバランスを正す

→offensive languageがhate speech として認識されやすくなる



TSNEによる可視化

30

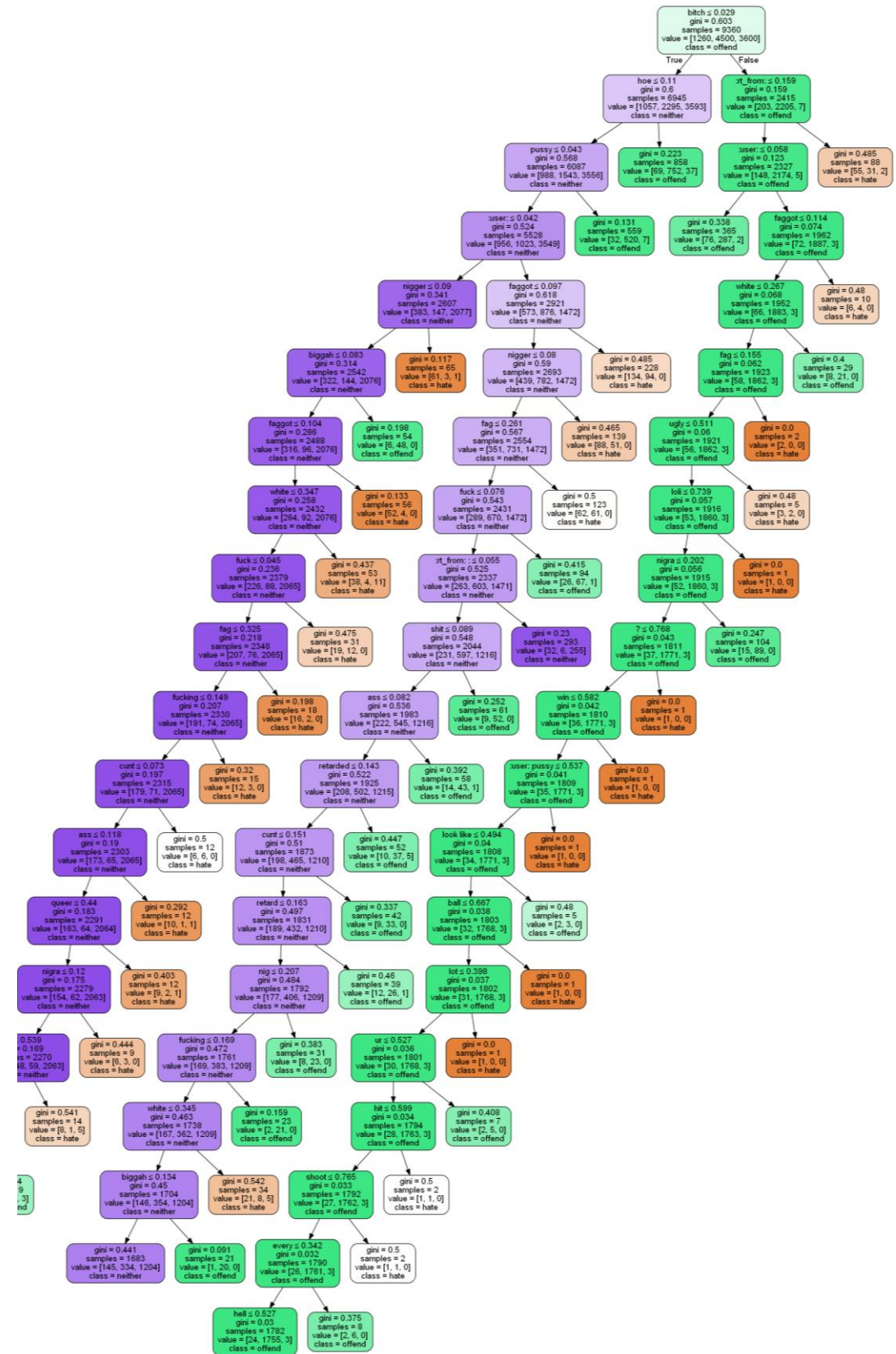


2つのクラスが重なり
あっている事が確認できる

- hate speech
- offensive language

決定木

ベストモデル



結論:ベストモデル

- モデルの判断の説明が簡単
 - ユーザーの投稿削除した時に理由が説明できる
 - ユーザー体験を損なわない
- 精度も遜色ない
 - f1-weighted 85%



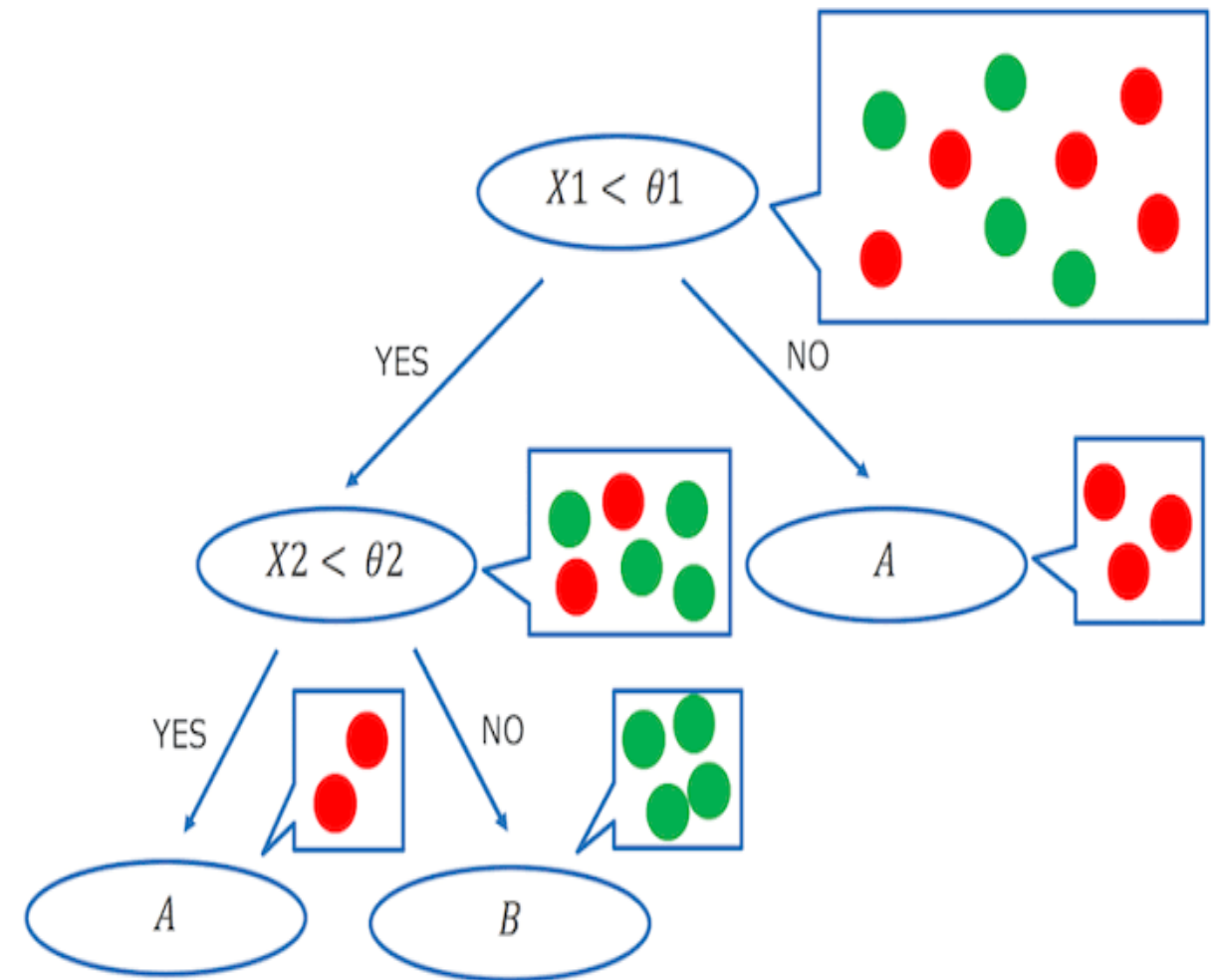
今後の課題

1. 正式なデータをもっと増やす
2. 決定木の欠陥の改善
3. SVM など他のモデルを試す
4. LSTM に関してはconvolutional 層を追加するなど
他の構成を試す

モデル紹介:補足

- 不純度が最も減少する特徴としきい値でノード二分割を繰り返す
- Gini係数を不純度として使用

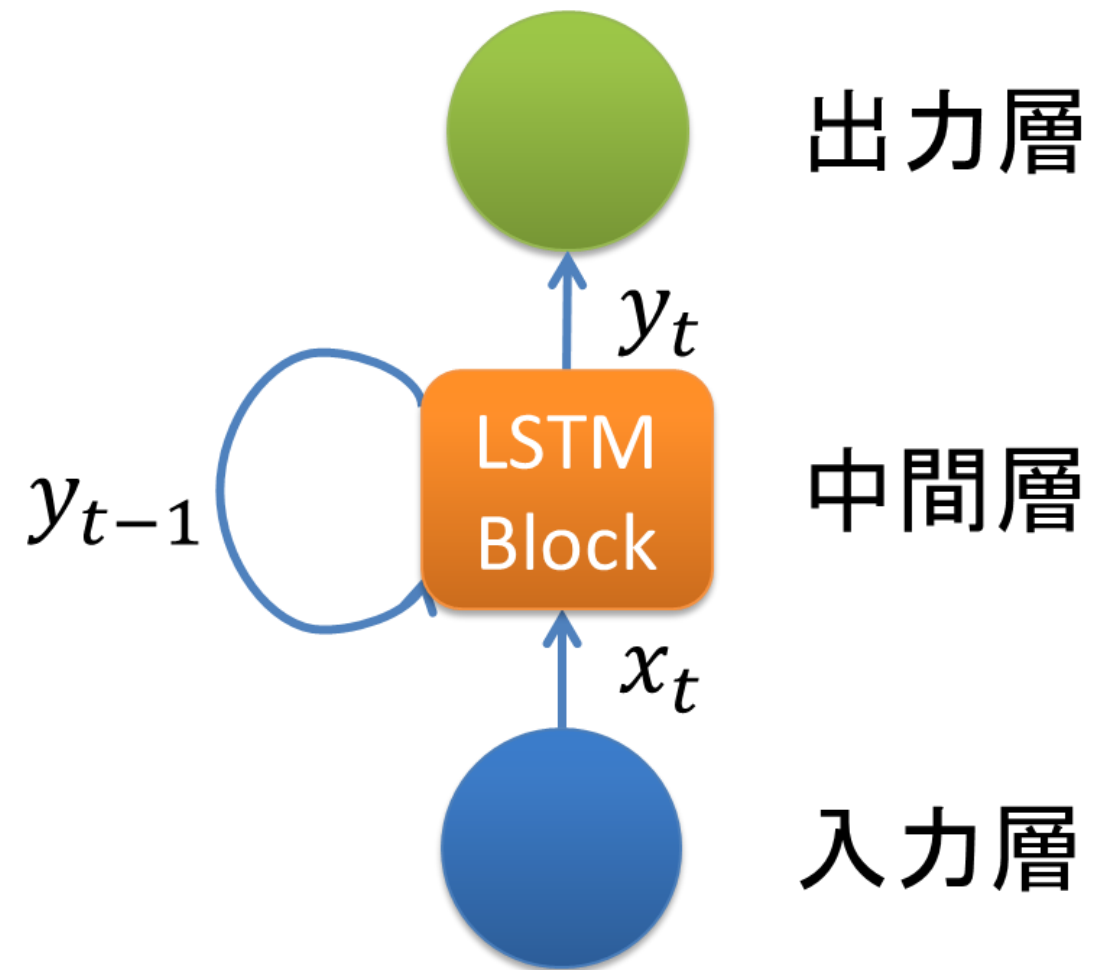
$$Gini = 1 - \sum_{i=1}^C (p_i)^2$$



モデル紹介:補足

35

- RNNの一種
- LSTM Block を中間層に使用以下の時系列(t-1) の情報を(t)へと再帰
 - $C(t-1)$:セル状態
 - $h(t-1)$: 出力
- 長期の依存性の学習が可能



Long-Short Term Memory