Kanazawa.RとKanzawa.R

Kanazawa.R & Kanzawa.R

23rd November 2024, Kanazawa.R #2

Yuta Kanzawa @yutakanzawa

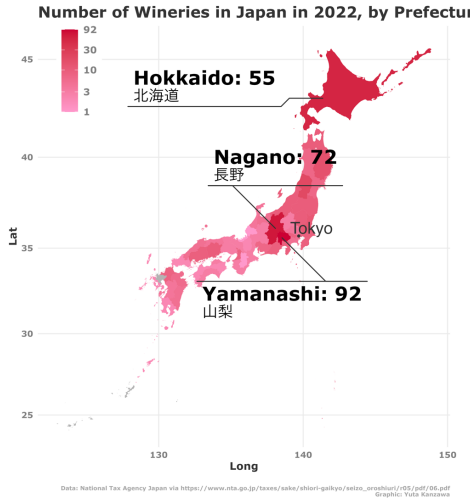
Data Scientist at Zurich Insurance Company Limited, Japan Branch

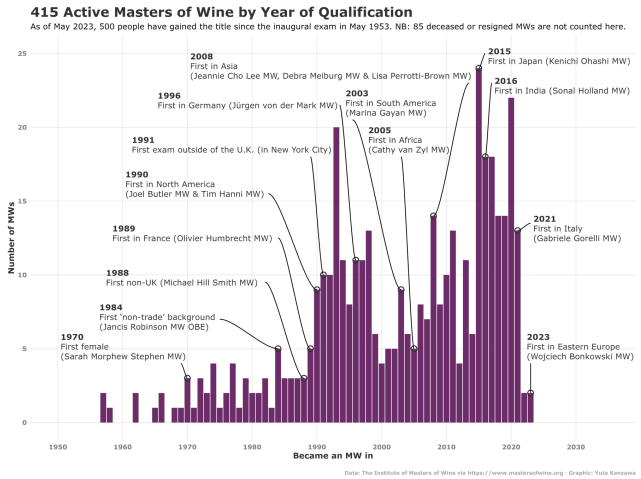
ZURICH

神沢雄大 Yuta Kanzawa

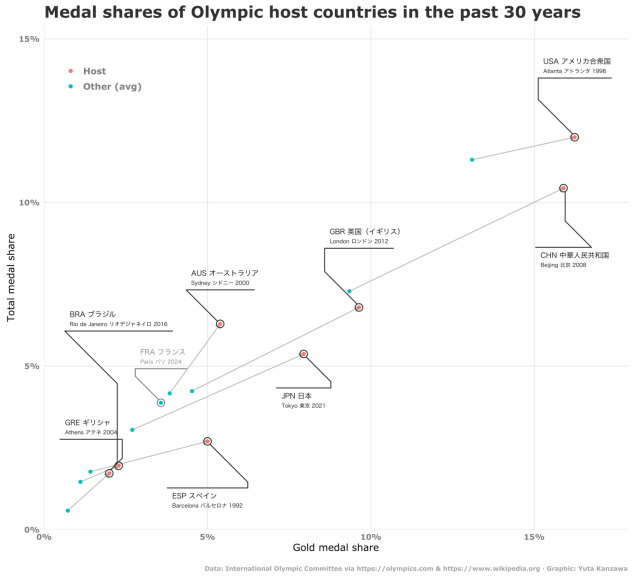
- データサイエンティスト@チューリッヒ保険会社
  - 日本支店
- Twitter: @yutakanzawa
- 好きなもの：オペラとワイン
  - ワーグナー
  - ブルゴーニュ (WSET Lv 3→?)
- 使用可能言語：7
  - 人間：日本語、英語、ドイツ語
  - コンピューター：R, Python, SAS, SQL

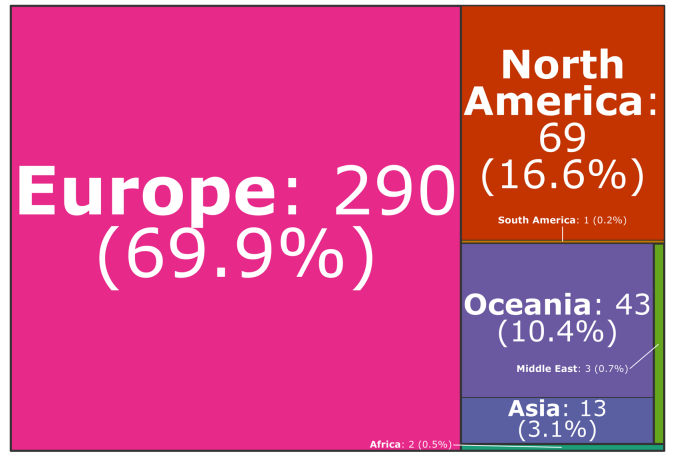
ポートフォリオ

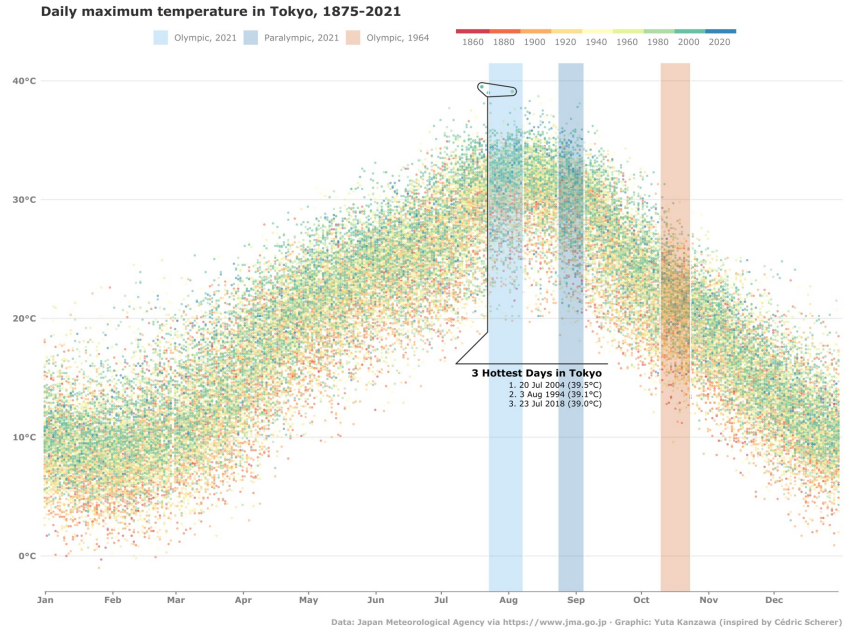




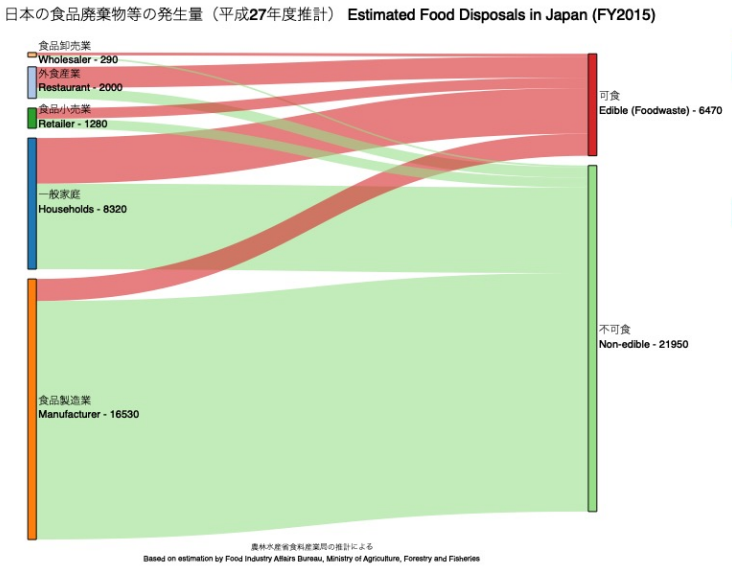
Source: Japan Sommelier Association https://www.sommelier.jp/exam/pdf/qualifiedholders.pdf
*Sommelier Excellence (2019-2020), Senior Sommelier (2013-2018), Senior Wine Adviser (2013-2015)

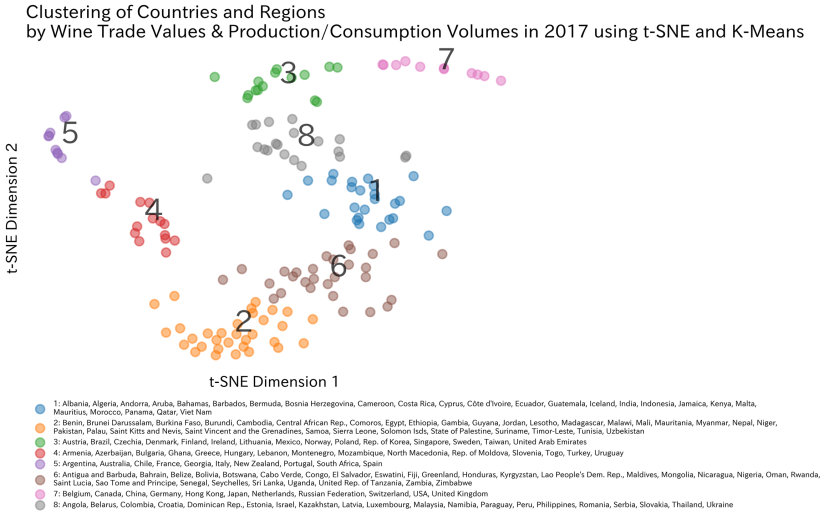


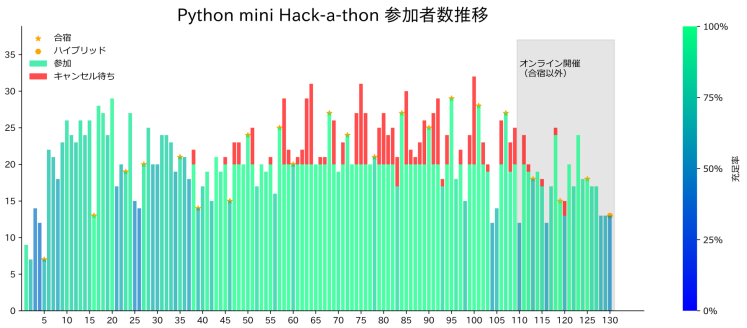


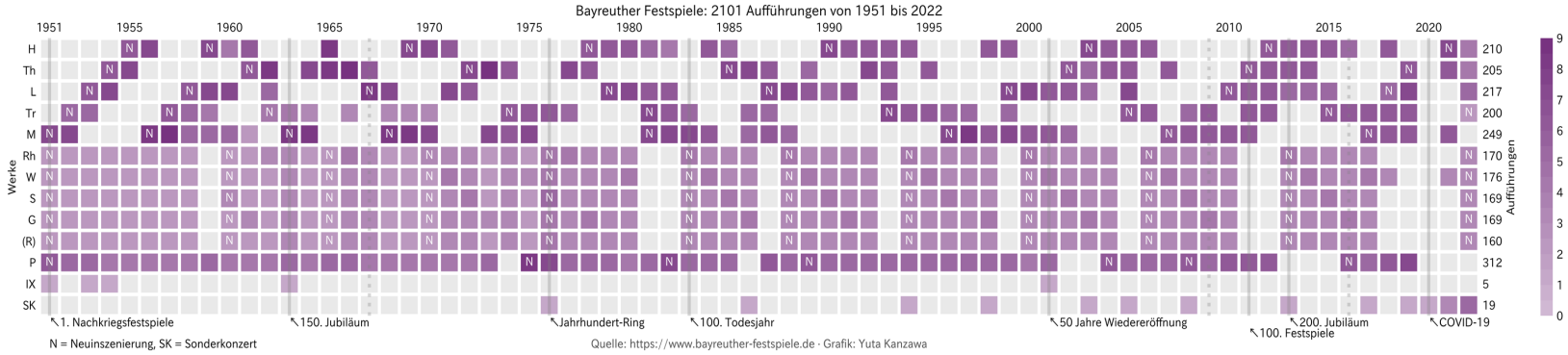


ポートフォリオ（参考までにR以外も）









アジェンダ

- 今日話すこと
  - 文字列同士の距離と類似度
  - stringdistパッケージ
- 対象（以下のいずれか）
  - 何らかの文字の配列を扱う人

- 今日話さないこと
  - Rでの実装

→文字列処理の豆知識

TL;DR

- 文字列同士の距離は主に次の3つの手法で評価される。
  - **編集距離**
    - ハミング距離、最長共通部分列距離、レーベンシュタイン距離、ダメラウ・レーベンシュタイン距離
  - **q-グラム**
    - q-グラム距離、ジャカール距離、コサイン距離
  - ヒューリスティクス（経験則）
    - **ジャロ距離**
- どれを使うかは用途次第だが、計算量と精度を考慮する。
- Rだと**stringdistパッケージ**で計算できる！

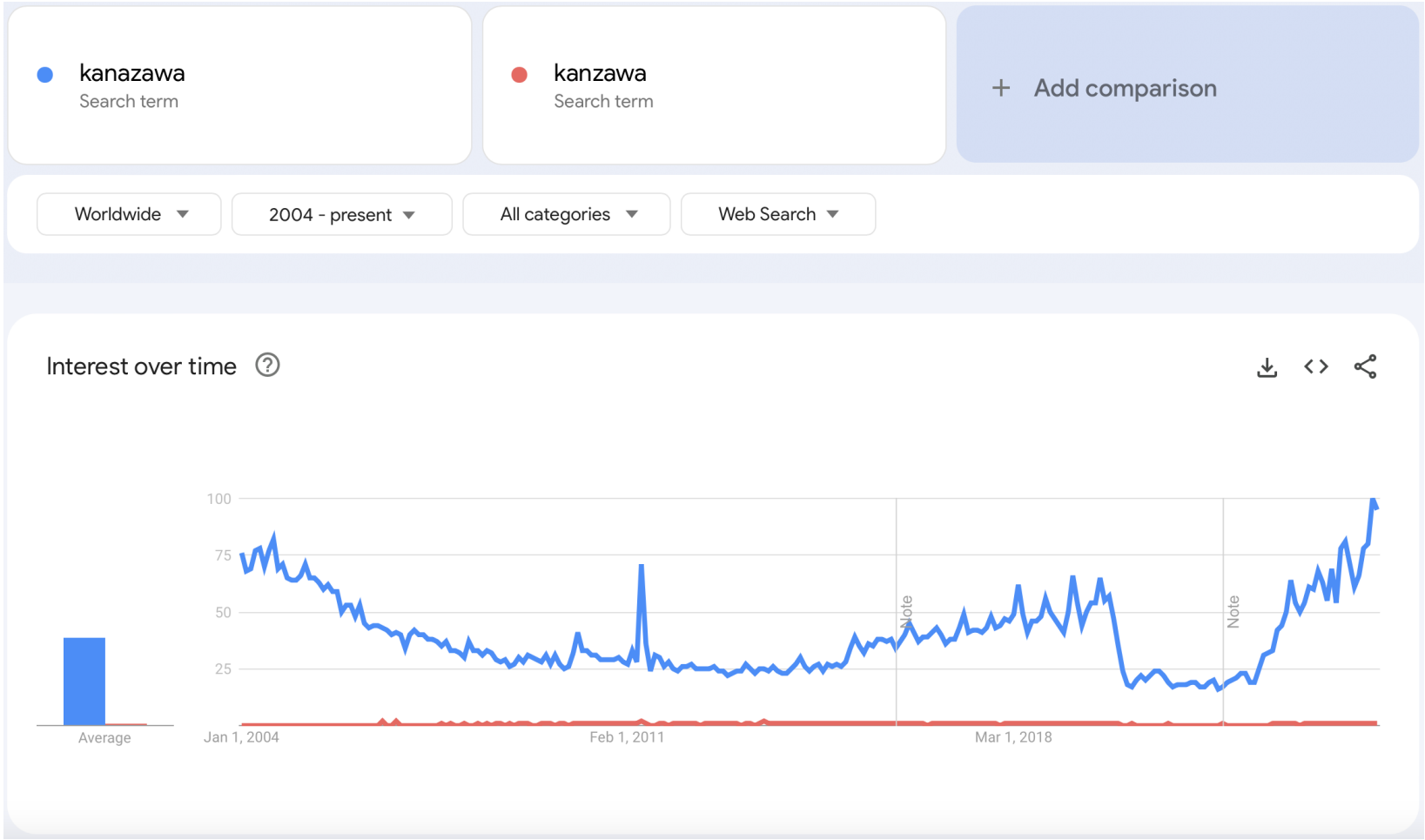
'Kanazawa' vs 'Kanzawa'

'Kanazawa' vs 'Kanzawa'

Googleの検索候補

* https://www.google.com

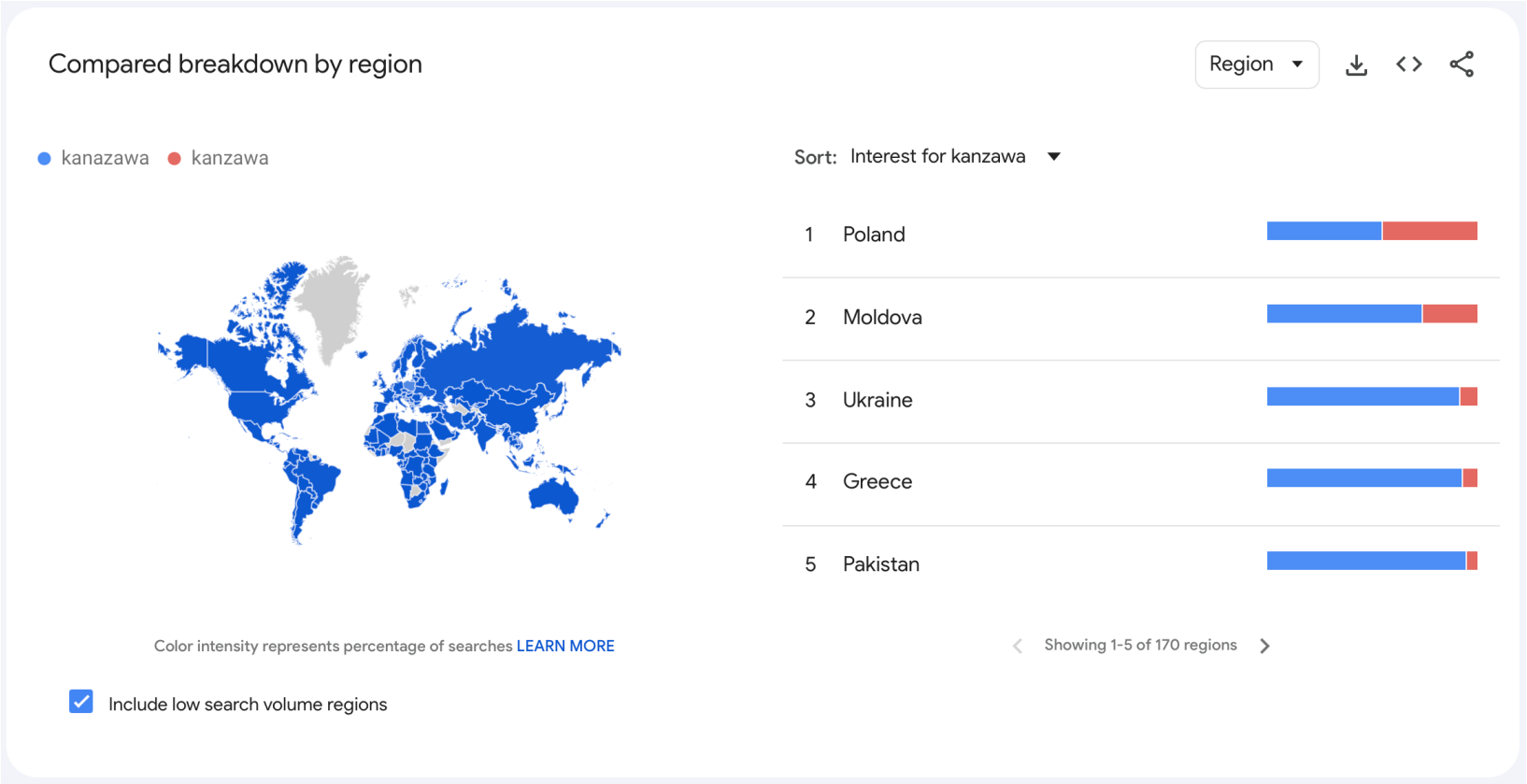
'Kanazawa' vs 'Kanzawa': 時系列



* https://trends.google.com/trends/explore?q=kanazawa,kanzawa&date=all

'Kanazawa' vs 'Kanzawa': 国別

Google Trends



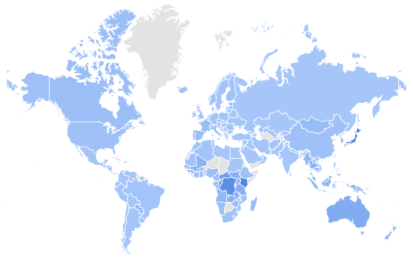
* https://trends.google.com/trends/explore?q=kanazawa,kanzawa&date=all

'Kanazawa' vs 'Kanzawa': 国別（詳細）

Google Trends

kanazawa

Interest by region | Region



Include low search volume regions

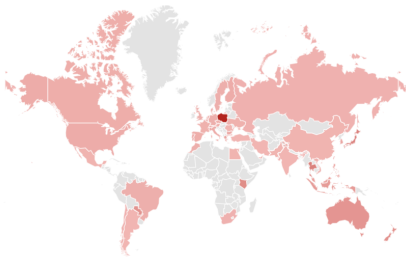
Related queries | Rising

1	tokyo to kanazawa	Breakout
2	kyoto to kanazawa	Breakout
3	shirakawago	Breakout
4	kanazawa shinkansen	Breakout
5	nagoya	Breakout

Showing 1-5 of 25 queries

kanzawa

Interest by region | Region



Include low search volume regions

Related queries | Rising

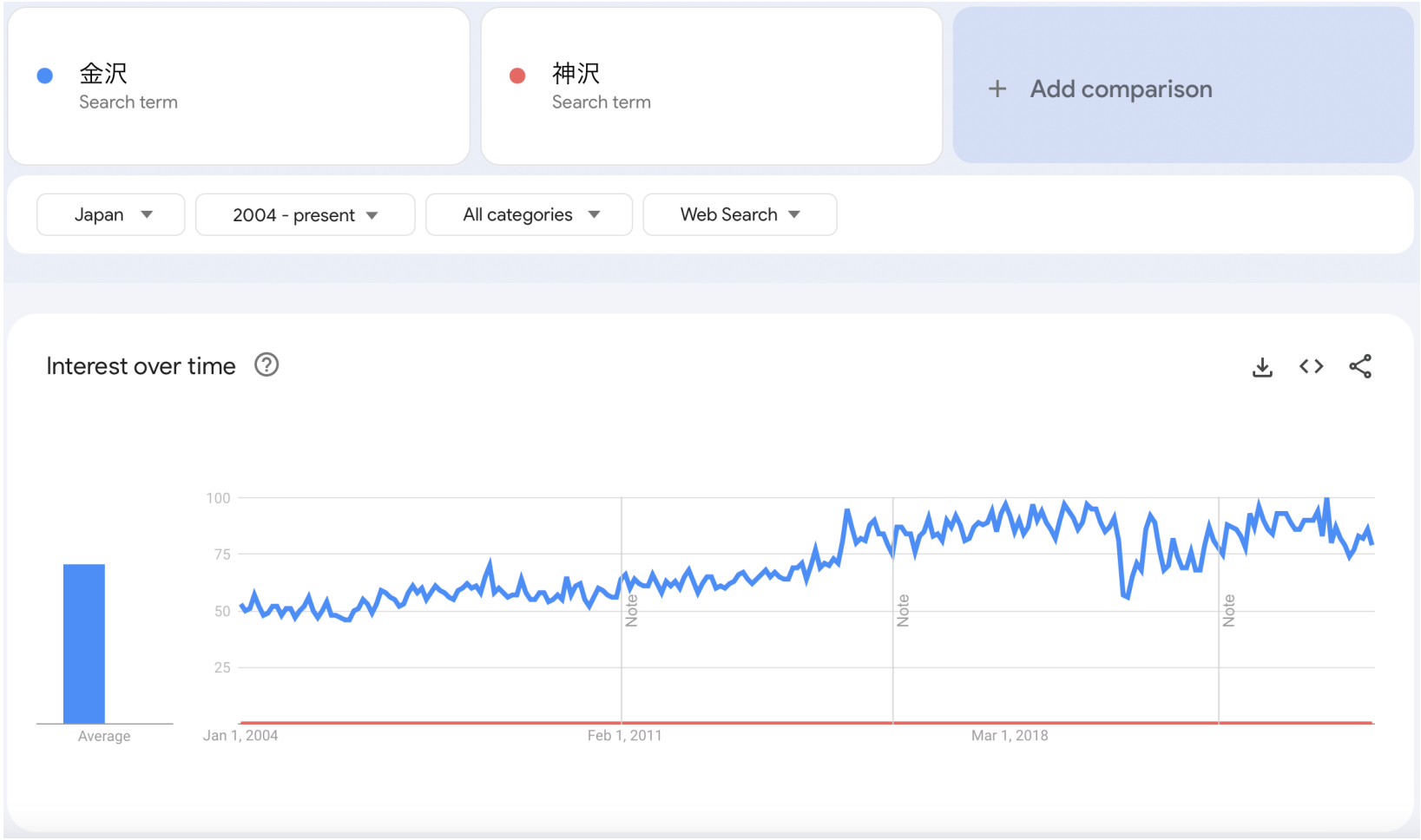
1	tokyo	Breakout
2	kyoto	Breakout
3	kanazawa japan	Breakout
4	kanzawa sensei	Breakout
5	takayama	Breakout

Showing 1-5 of 12 queries

* https://trends.google.com/trends/explore?q=kanazawa,kanzawa&date=all

「金沢」vs「神沢」：時系列

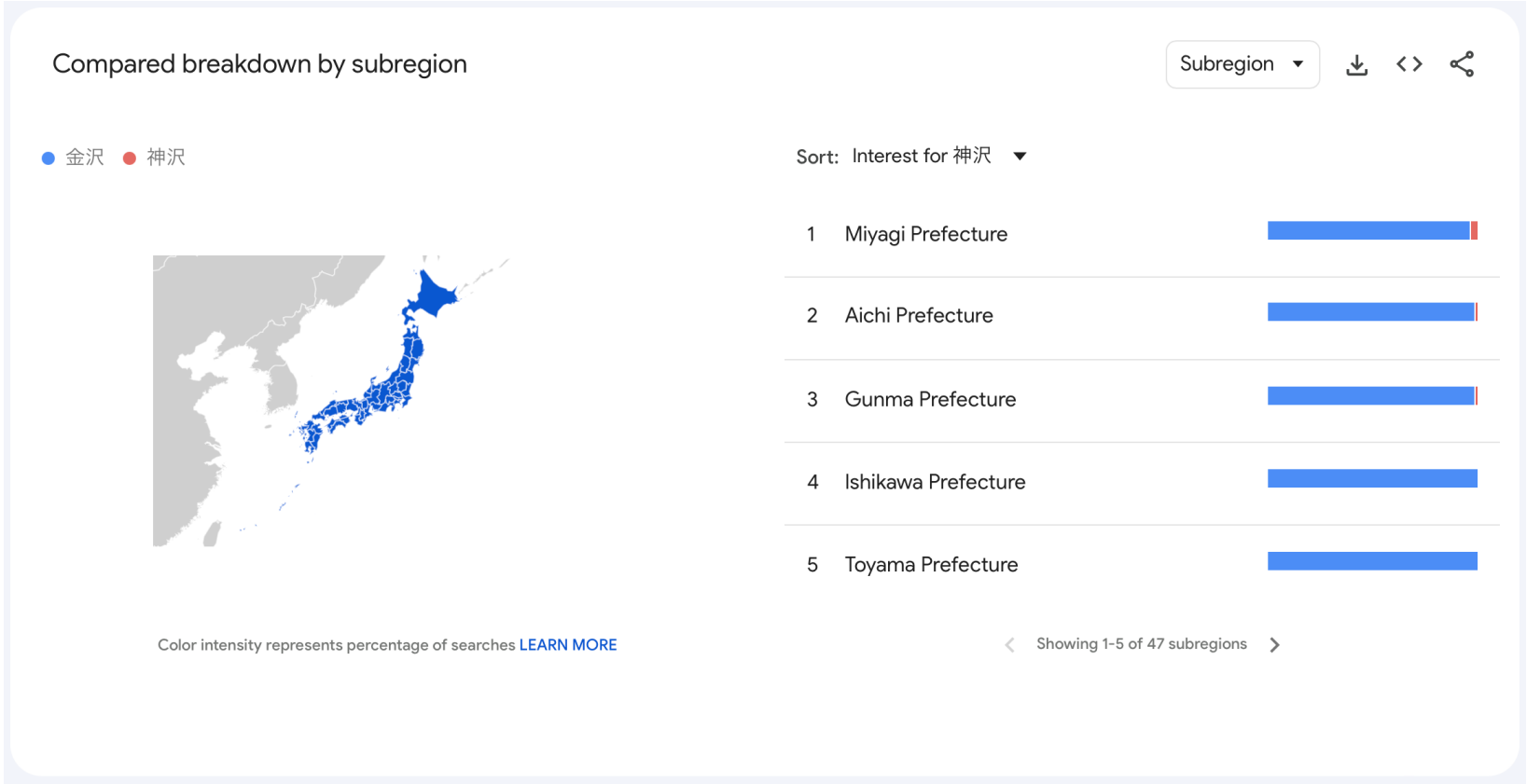
Google Trends



* https://trends.google.com/trends/explore?date=all&geo=JP&q=金沢,神沢

「金沢」vs「神沢」：都道府県別

Google Trends



* https://trends.google.com/trends/explore?date=all&geo=JP&q=金沢,神沢

「金沢」vs「神沢」：都道府県別（詳細）

Google Trends

金沢

Interest by subregion　Subregion

Related queries　Rising

1	フォーラス 金沢	Breakout
2	ツエーゲン 金沢	Breakout
3	フォーラス 映画	Breakout
4	金沢 フォーラス 映画	Breakout
5	名古屋 から 金沢	Breakout

Showing 1-5 of 25 queries

神沢

Interest by subregion　Subregion

Related queries　Rising

1	仙台 市 泉 区	Breakout
2	仙台 市 泉 区 天 神沢	Breakout
3	地下鉄	Breakout
4	神沢 皮膚 科	Breakout
5	神沢 精工	Breakout

Showing 1-5 of 12 queries

* https://trends.google.com/trends/explore?date=all&geo=JP&q=金沢,神沢

文字列同士の距離と類似度

String distances & similarities

おことわり

- ここからは主に次のstringdistパッケージの論文の内容を紹介します。
  - van der Loo, Mark P.J. (2014), “The stringdist Package for Approximate String Matching”, The R Journal 6 (1): 111-122
  - 説明の都合上、順序や記法を変えているところがあります。
- 原論文に由来するもの以外の誤りは全て発表者に帰するものです。

「距離」の概念

- 2つの文字列がどのくらい「離れている」かを定量的に表すもの。
  - Cf.「類似度」：どのくらい似ているか
    - 距離 ＝ 1 − 類似度
  - 距離関数の持つべき性質：非負性、同一性、対称性、三角不当性
- 主なもの
  - 編集距離
  - q-グラム
  - ヒューリスティクス（経験則）

編集距離

Edit distances

編集距離

- ある文字列を**別の文字列に変換する操作の最小回数**
  - 置換：任意の1文字を**他の1文字で置き換える**。**a**bc → **x**bc
  - 削除：任意の1文字を**取り除く**。a**b**c → ac
  - 挿入：1文字を**任意の位置に加える**。abc → a**y**bc
  - 転置（交換）：隣り合う2文字の**位置を入れ替える**。a**bc** → a**cb**
- 主なもの
  - ハミング距離：置換
  - 最長共通部分列距離：削除、挿入
  - レーベンシュタイン距離：置換、削除、挿入
  - ダメラウ・レーベンシュタイン距離：置換、削除、挿入、転置

ハミング距離

- Hamming, 1950
- ある文字列を別の文字列に変換するのに必要な**置換**の**最小**回数
  - 原則として**同じ長さ**の文字列同士
  - 2つの文字列の間で同じ位置にある異なる文字の数と同義。
    - →**長距離通信のエラー検出**
- 例：「かなざわ」と「かんざわ」→ 1（置換1回）
  - か**な**ざわ → か**ん**ざわ

* Hamming, R. (1950), “Error detecting and error correcting codes.”, The Bell system technical journal 29: 147–160

最長共通部分列距離

- Needleman & Wunsch, 1970
- **最長共通部分列に含まれない文字の数**
  - 最長共通部分列：2つの文字列に同じ順序で出現する文字の列
    - 連続している必要はない。
  - ある文字列を別の文字列に変換するのに必要な**削除と挿入**の**最小**回数
  - 発展形 → スミス・ウォーターマン法（Smith & Waterman, 1981）
- **類似したDNA配列の探索**のために考案された。
- 例：「かなざわ」と「かんざわ」→ 2
  - 最長共通部分列=「かざわ」→ 残り：「な」、「ん」
  - か**な**ざわ → かざわ → か**ん**ざわ（削除1回、挿入1回）

* Needleman, S; Wunsch, C. D. (1970), “A general method applicable to the search of similarities in the amino acid sequence of two proteins”, Journal of Molecular Biology 48: 443–453

レーベンシュタイン距離

- Levenshtein, 1965
- ある文字列を別の文字列に変換するのに必要な**置換、削除、挿入**の**最小**回数
  - 文字列同士の長さは異なっていてもOK。
  - ハミング距離はレーベンシュタイン距離の特殊なケース
- 例：「かなざわ」と「いしかわけん」→ 5（置換3回、挿入2回）
  - **か**なざわ → **い**なざわ → い**し**ざわ → いし**か**わ → いしかわ**け** → いしかわ**ん**

* Levenshtein, V. I. (1965), “Binary codes capable of correcting deletions, insertions, and reversals”, Doklady Akademii Nauk SSSR 163 (4): 845–848

ダメラウ･レーベンシュタイン距離

- Damerau, 1964; Lowrance & Wagner, 1975
- ある文字列を別の文字列に変換するのに必要な**置換、削除、挿入、転置（交換）**の**最小**回数
  - 交換は隣接する2文字間のみ。
  - **スペルチェッカー、DNA解析**
- 例：「かなざわ」と「なかざわ」→ 1（転置1回）
  - **かな**ざわ → **なか**ざわ
  - レーベンシュタイン距離は2（置換2回）
    - **か**なざわ → **な**なざわ → な**か**ざわ

* Damerau, Fred J. (1964), “A technique for computer detection and correction of spelling errors”, Communications of the ACM 7 (3): 171–176
* Lowrance, Roy; Wagner, R. (1975), “An Extension of the String-to-String Correction Problem”, Journal of the Association of Computing Machinery 22 (2): 177–183

q-グラム

q-gram

q-グラム

- **連続**するq文字の**部分文字列**
  - 例：「かなざわ」
    - 1-グラム（q=1）：「か」、「な」、「ざ」、「わ」
    - 2-グラム（q=2）：「かな」、「なざ」、「ざわ」
  - →文字列を**q-グラムの集合やベクトル**で表し、距離や類似度を評価。
    - ただし、1グラムは文字の順序が考慮されないので、**アナグラムは距離0**となる。
- 主なもの
  - q-グラム距離
  - q-グラムのジャカール距離
  - q-グラムのコサイン距離

q-グラム距離（1/2）

- Ukkonen, 1992
- q-グラムの多重集合（要素の重複あり）の**差集合の要素数**
  - q=1の場合は、文字列間で対応しない文字の数
  - **各文字列のq-グラムの出現頻度のベクトルの差分の1-ノルム**と同義。
- 例1：「かなざわ」と「かんざわ」→ 2 (q=1), 4 (q=2)
  - q=1: {{**か**,な,**ざ**,**わ**}}, {{**か**,ん,**ざ**,**わ**}} → {{な, ん}}
    - (か,ざ,な,わ,ん) ← 各文字の出現頻度のベクトル
    - $\|(\mathbf{1},\mathbf{1},1,\mathbf{1},0)-(\mathbf{1},\mathbf{1},0,\mathbf{1},1)\|_1 = \|(0,0,1,0,-1)\|_1 = |1|+|-1| = 2$
  - q=2: {{かな,なざ,**ざわ**}}, {{かん,んざ,**ざわ**}} → {{かな,なざ,かん,んざ}}

* Ukkonen, E. (1992), “Approximate string-matching with q-grams and maximal matches”, Theoretical Computer Science 92: 191–211

q-グラム距離（2/2）

- 例2：「かなざわ」と「かが」→ 4 (q=1), 4 (q=2)
  - q=1: {{**か**,な,ざ,わ}}, {{**か**,が}} → {{な,ざ,わ,が}}
  - q=2: {{かな,なざ,ざわ}}, {{かが}} → {{かな,なざ,ざわ,かが}}
- 例3：「かなざわ」と「たかおか」→ 6 (q=1)
  - q=1: {{**か**,な,ざ,わ}}, {{た,**か**,お,**か**}} → {{な,ざ,わ,た,お,**か**}}
- 例4：「かなざわ」と「とやま」→ 7 (q=1)
  - q=1: {{か,な,ざ,わ}}, {{と,や,ま}} → {{か,な,ざ,わ,と,や,ま}}
- 例5：「かなざわ」と「なかざわ」→ 0（q=1）…アナグラムの場合
  - q=1: {{**か**,**な**,**ざ**,**わ**}}, {{**な**,**か**,**ざ**,**わ**}} → {{}}

q-グラムのジャカール距離（1/2）

- Jaccard, 1901
- **1 – ジャカール指数**
  - ジャカール指数（係数）：2つの集合の類似度を**0から1**の値で表す。
    - **2つの集合の積集合の要素数（濃度）の、和集合の要素数に対する比**：$\frac{|Q_1 \cap Q_2|}{|Q_1 \cup Q_2|}$
  - 全く同じ文字がないとq=1でもジャカール距離は1（最遠）となる。
- 例1：「かなざわ」と「かんざわ」→ 0.4 (q=1), 0.8 (q=2)
  - q=1: Q_1={**か**,な,**ざ**,**わ**}, Q_2={**か**,ん,**ざ**,**わ**} → $\frac{|Q_1 \cap Q_2|}{|Q_1 \cup Q_2|} = \frac{3}{5} = 0.6$
  - q=2: Q_1={かな,なざ,**ざわ**}, Q_2={かん,んざ,**ざわ**} → $\frac{|Q_1 \cap Q_2|}{|Q_1 \cup Q_2|} = \frac{1}{5} = 0.2$

* Jaccard, Paul. (1901), ”Étude comparative de la distribution florale dans une portion des Alpes et des Jura”, Bulletin de la Société vaudoise des sciences naturelles (in French). 37 (142): 547–579

q-グラムのジャカール距離（2/2）

- 例2：「かなざわ」と「かが」→ 0.8 (q=1), 1 (q=2)
  - q=1: Q_1= {**か**,な,ざ,わ}, Q_2={**か**,が} → $\frac{|Q_1 \cap Q_2|}{|Q_1 \cup Q_2|} = \frac{1}{5} = 0.2$
  - q=2: Q_1={かな,なざ,ざわ}, Q_2={かが} → $\frac{|Q_1 \cap Q_2|}{|Q_1 \cup Q_2|} = \frac{0}{4} = 0$
- 例3：「かなざわ」と「たかおか」→ 0.83 (q=1)
  - q=1: Q_1={**か**,な,ざ,わ}, Q_2={た,**か**,お} → $\frac{|Q_1 \cap Q_2|}{|Q_1 \cup Q_2|} = \frac{1}{6} = 0.1666 \ldots$
- 例4：「かなざわ」と「とやま」→ 1 (q=1)
- 例5：「かなざわ」と「なかざわ」→ 0（q=1）
  - q=1: Q_1= {**か**,**な**,**ざ**,**わ**}, Q_2={**な**,**か**,**ざ**,**わ**} → $\frac{|Q_1 \cap Q_2|}{|Q_1 \cup Q_2|} = \frac{4}{4} = 1$

q-グラムのコサイン距離（1/2）

- コサイン距離 ＝ 1 － **コサイン類似度**
  - コサイン類似度：**2つのベクトルの内積の、大きさの積に対する比：** $\frac{a \cdot b}{|a||b|}$
- **q-グラムの出現頻度のベクトル同士のコサイン距離**
  - 出現頻度は0以上なので、q-グラムのコサイン距離は**0から1**の値をとる。
  - 全く同じ文字がないとコサイン距離は1（最遠）となる。
- 例1：「かなざわ」と「かんざわ」→ 0.25 (q=1)
  - q=1:「かなざわ」→ (か,ざ,な,わ)、「かんざわ」→ (か,ざ,わ,ん)
    - (か,ざ,な,わ,ん) ← 各文字の出現頻度のベクトル
    - $\boldsymbol{v}_1 = (1,1,1,1,0), \boldsymbol{v}_2 = (1,1,0,1,1)$ → $\frac{\boldsymbol{v}_1 \cdot \boldsymbol{v}_2}{|\boldsymbol{v}_1||\boldsymbol{v}_2|} = \frac{3}{2 \cdot 2} = 0.75$（コサイン類似度）

q-グラムのコサイン距離（2/2）

- 例2：「かなざわ」と「かが」→ 0.65 (q=1)
  - q=1:「かなざわ」→ (か,ざ,な,わ)、「かが」→ (か,が)
    - (か,が,ざ,な,わ,) ← 各文字の出現頻度のベクトル
    - $\boldsymbol{v}_1 = (1,0,1,1,1), \boldsymbol{v}_2 = (1,1,0,0,0)$ → $\frac{\boldsymbol{v}_1 \cdot \boldsymbol{v}_2}{|\boldsymbol{v}_1||\boldsymbol{v}_2|} = \frac{1}{2 \cdot \sqrt{2}} = 0.353 \ldots$
- 例3：「かなざわ」と「たかおか」→ 0.59 (q=1)
  - q=1:「かなざわ」→ (か,ざ,な,わ)、「たかおか」→ (お,か,た)
    - (お,か,ざ,た,な,わ) ← 各文字の出現頻度のベクトル
    - $\boldsymbol{v}_1 = (0,1,1,0,1,1), \boldsymbol{v}_2 = (1,2,0,1,0,0)$ → $\frac{\boldsymbol{v}_1 \cdot \boldsymbol{v}_2}{|\boldsymbol{v}_1||\boldsymbol{v}_2|} = \frac{2}{2 \cdot \sqrt{6}} = 0.408 \ldots$
- 例4：「かなざわ」と「とやま」→ 1 (q=1)

ヒューリスティクス（経験則）

Heuristics

ヒューリスティクス（経験則）

- ある文字列が本来あるべき真の文字列から誤りによって生まれる場合、その原因を経験的に仮定し、距離の計算方法に反映。
  - 位置の**近い文字同士の違い**は**タイプミス**によって起こる。
  - 位置が遠い場合の違いはそうではない。

- **ジャロ距離**
  - 米国統計局にて調査時の**誤字脱字**を処理するために考案（1978年）。
  - 上記の仮定に基づき、2つの文字列の類似度を算出。
  - 拡張：**ジャロ・ウィンクラー距離**

ジャロ距離（1/4）

- Jaro, 1978; Jaro, 1989
- 定義（文字列s_1とs_2のジャロ**類似度**）：
  - $\frac{1}{3}\left(w_1\frac{m}{|s_1|}+w_2\frac{m}{|s_2|}+w_3\frac{m-T}{m}\right)$ → 1からこれを引いたものがジャロ距離
    - w_1, w_2, w_3：重み（通常は1）
    - $|s_1|, |s_2|$：文字列の長さ（文字数）
    - m：**近傍で一致する文字の数**（位置が同じか近い同じ文字）
      - 比較する範囲：$\left\lfloor\frac{\max(|s_1|,|s_2|)}{2}\right\rfloor - 1$文字以内（0なら同じ位置のみ）
    - T：mで「一致」した文字のうち**転置の数**（位置が入れ替わっている文字ペアの数）
      - 「一致」しなかった文字を除いた文字列同士の比較
      - 隣接していなくてもよい。
    - ある1文字を**使うのは一度だけ**。

* Jaro, M. (1978), “UNIMATCH: A record linkage system: User manual”, United States bureau of the census: 103-108
* Jaro, M. (1989), “Advances in record-linkage methodology as applied to matching the 1985 census of Tampa, Florida”, Journal of the American Statistical Association 84: 414–420

ジャロ距離（2/4）

- 例1：「かなざわ」と「かんざわ」→ 0.17
  - 重みは全て1とする。
  - $|s_1| = 4, |s_2| = 4$
  - 一致の範囲：$\left\lfloor \frac{\max(|s_1|,|s_2|)}{2} \right\rfloor - 1 = \left\lfloor \frac{\max(4,4)}{2} \right\rfloor - 1 =$ 1文字以内
  - $m = 3$（**か**な**ざわ**,**か**ん**ざわ**）
  - $T = 0$（転置なし）
  - ジャロ類似度：$\frac{1}{3}\left(\frac{m}{|s_1|} + \frac{m}{|s_2|} + \frac{m-T}{m}\right) = \frac{1}{3}\left(\frac{3}{4} + \frac{3}{4} + \frac{3-0}{3}\right) = \frac{5}{6} = 0.8333 \ldots$

ジャロ距離（3/4）

- 例2：「かなざわ」と「たかおか」→ 0.5
  - 重みは全て1とする。
  - $|s_1| = 4, |s_2| = 4$
  - 一致の範囲：$\left\lfloor \frac{\max(|s_1|,|s_2|)}{2} \right\rfloor - 1 = \left\lfloor \frac{\max(4,4)}{2} \right\rfloor - 1 =$ 1文字以内
  - $m = 1$（**か**なざわ,た**か**おか）
  - $T = 0$（転置なし）
  - ジャロ類似度：$\frac{1}{3}\left(\frac{m}{|s_1|} + \frac{m}{|s_2|} + \frac{m-T}{m}\right) = \frac{1}{3}\left(\frac{1}{4} + \frac{1}{4} + \frac{1-0}{1}\right) = \frac{1}{2} = 0.5$

ジャロ距離（4/4）

- 例3：「かなざわ」と「なかざわ」→ 0.08
  - 重みは全て1とする。
  - $|s_1| = 4, |s_2| = 4$
  - 一致の範囲：$\left\lfloor \frac{\max(|s_1|,|s_2|)}{2} \right\rfloor - 1 = \left\lfloor \frac{\max(4,4)}{2} \right\rfloor - 1 =$ 1文字以内
    → $m = 4$（**かなざわ,なかざわ**）
  - $T = 1$（「か」と「な」が入れ替わっている）
  - ジャロ類似度：$\frac{1}{3}\left(\frac{m}{|s_1|} + \frac{m}{|s_2|} + \frac{m-T}{m}\right) = \frac{1}{3}\left(\frac{4}{4} + \frac{4}{4} + \frac{4-1}{4}\right) = \frac{11}{12} = 0.91666\ldots$

どれを使うべきか？

- 用途次第
  - ジャロ距離は元々実用的な目的のために考案された。
- 考慮すべき点
  - **計算量**（評価する文字列の長さ）
    - q-グラム < 編集距離
  - **精度**
    - 編集距離 > q-グラム

応用例

- 通信エラーの検出
- 誤りの訂正候補
- 曖昧検索
  - 実務的にはより高度なアルゴリズムが使われることが多い。
- DNA配列の探索

stringdistパッケージ

The stringdist package

stringdistパッケージ早見表

- stringdist()関数
  - **stringdist("文字列1", "文字列2", method = "手法")**

種類	指標名	引数method	備考
編集距離	ハミング距離	"hamming"	
	最長共通部分列距離	"lcs"	
	レーベンシュタイン距離	"lv"	
	ダメラウ・レーベンシュタイン距離	"dl"	
q-グラム	q-グラム距離	"qgram"	文字数は引数qに渡す。
	ジャカール距離	"jaccard"	
	コサイン距離	"cosine"	
ヒューリスティクス	ジャロ距離	"jw"	ジャロ・ウィンクラー距離は引数pにプレフィクススケールを渡す。

「Kanazawa.R」と「Kanzawa.R」の距離

種類	指標名	値	備考
編集距離	ハミング距離	-	長さが違うため計算不可。
	最長共通部分列距離	1	
	レーベンシュタイン距離	1	
	ダメラウ・レーベンシュタイン距離	1	
q-グラム	q-グラム距離	1	
	ジャカール距離	0	
	コサイン距離	0.009	
ヒューリスティクス	ジャロ距離	0.107	

まとめ

Long story short

まとめ

- 文字列同士の距離は主に次の3つの手法で評価される。
  - **編集距離** → 高精度
    - ハミング距離、最長共通部分列距離、レーベンシュタイン距離、ダメラウ・レーベンシュタイン距離
  - **q-グラム** → 低計算量
    - q-グラム距離、ジャカール距離、コサイン距離
  - ヒューリスティクス（経験則）
    - **ジャロ距離**
- どれを使うかは用途次第だが、計算量と精度を考慮する。
  - 今日紹介した以外にも新たなアルゴリズムが考案されている。
- Rだと**stringdistパッケージ**で計算できる！

Enjoy!