

分散型 SNS におけるユーザの潜在要求分析

プロジェクトマネジメントコース

ソフトウェア開発管理グループ

矢吹研究室

1442037

加藤 健弥

目次

第 1 章	序論	2
第 2 章	背景	3
第 3 章	目的	4
第 4 章	手法	5
4.1	研究の対象にする Mastodon のインスタンス	6
4.2	研究に必要な道具の準備	7
4.3	研究に使うデータの取得	19
4.4	つぶやきの主成分分析	26
第 5 章	結果	29
第 6 章	考察	59
第 7 章	結論	60
	参考文献	61
	謝辞	62

第 1 章

序論

本研究では，分散型 SNS と中央集権型 SNS を定量的に調査，分析を行う．分散型 SNS は中央集権型 SNS と求められていることに違いがあるのかを調査することとした．

第 2 章

背景

スマートフォンなどの普及により，手軽にインターネットへの接続が可能になった．そのため，Twitter や Facebook などの様々な SNS（ソーシャルネットワークサービス）が注目されるようになった．近年では Mastodon という新たな SNS の利用者が増えてきている．

Mastodon とは 2016 年に公開されたオープンソースソフトウェアであり，誰でも自由にサーバを立てて運用できる．そのため，Twitter や Facebook のような利用者が一つのサーバにログインする中央集権型のサービスに対して Mastodon の利用者は管理者も設置場所も異なるサーバにあるインスタンスにログインする分散型のサービスである [1]．

インスタンスとは，Mastodon を運用しているそれぞれのサーバのことである．そのため，利用者は別のインスタンスの利用者とはつながっていない．しかしインスタンス同士が連合という形で結びつくことができるため，別のインスタンスであっても連合であれば利用者同士でつながることができる [2]．

第 3 章

目的

Twitter と Mastodon で，投稿される話題に違いがあるかを，つぶやきを定量的に分析することによって調査する．

第 4 章

手法

全体のながれとしては以下のとおりである。

1. Twitter の API キー，アクセストークンを取得する。
2. Mastodon の API キー，アクセストークンを取得する。
3. Twitter API を使用し，Twitter から無作為に 100 のつぶやきを取得する。
4. Mastodon API を使用し，Mastodon の 30 のインスタンスから 1 つのインスタンスごとに無作為に 100 のつぶやきを取得する。
5. Twitter と Mastodon のつぶやきを Word2vec によってベクトル化する。
6. Twitter と Mastodon の各インスタンス同士で主成分分析する。

4.1 研究の対象にする Mastodon のインスタンス

研究の対象にする Mastodon のインスタンスは以下のとおりである。

1. aidon.club
2. bicyclemstdn.jp
3. catdon.life
4. dq10.online
5. eigadon.net
6. fgochiho.vip
7. friends.nico
8. gamecreate.mstdn.cloud
9. ika.queloud.net
10. imastodon.net
11. kero.ccsakura.jp
12. kirakiratter.com
13. konkat.jp
14. kurage.cc
15. mast.moe
16. mastodon.bitbank.cc
17. mastodon.cosmicanimal.jp
18. mastodon.fishing
19. mastodon.yokohama
20. mstdn.hokkaido.jp
21. mstdn.jp
22. mstdn.osaka
23. mstdn-football.jp
24. mstdn-kanazawa.jp
25. now.kibousoft.co.jp
26. pawoo.net
27. ro-mastodon.puyo.jp
28. toot.redmine.jp
29. tuner.1242.com
30. vocalodon.net

4.2 研究に必要な道具の準備

4.2.1 Chocolatey の導入

Chocolatey とは

Chocolatey とはコマンドラインで操作をすることができる Windows 用パッケージマネージャーである。LinuxOS のパッケージ管理コマンドのように使うことができる。

Chocolatey のインストール

以下のようにホームページからインストールする。

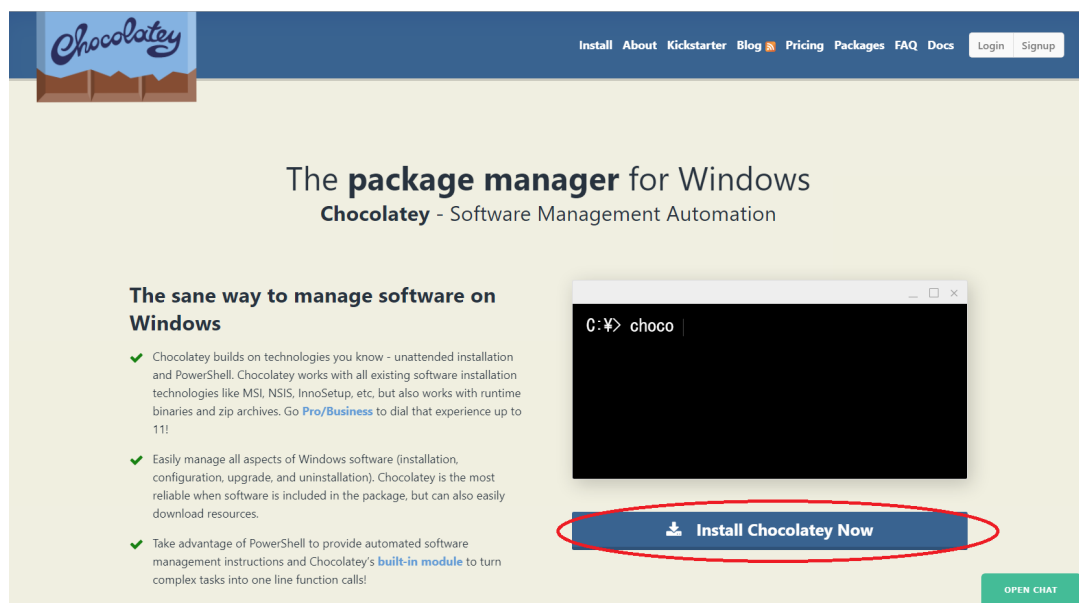


図 4.1 Chocolatey のホームページ

公式ホームページのインストールページに行き，Install Chocolatey Now を選択する。

Installing Chocolatey


Chocolatey installs in seconds. You are just a few steps from running `choco` right now!

1. First, ensure that you are using an [administrative shell](#) - you can also install as a non-admin, check out [Non-Administrative Installation](#).
2. Copy the text specific to your command shell - [cmd.exe](#) or [powershell.exe](#).
3. Paste the copied text into your shell and press Enter.
4. Wait a few seconds for the command to complete.
5. If you don't see any errors, you are ready to use Chocolatey! Type `choco` or `choco -?` now.

NOTES:

If you are behind a proxy, please see [Installing behind a proxy](#).
Need completely offline solution? See [Completely Offline Install](#).
Installing the licensed edition? See [install licensed edition](#).
[More Options / Troubleshooting](#)

Install with cmd.exe


Run the following command:  (copy command text)

```
@'%SystemRoot%\System32\WindowsPowerShell\v1.0\powershell.exe' -NoProfile -InputFormat None -ExecutionPolicy Bypass -Command "iex ((New-Object System.Net.WebClient).DownloadString('https://chocolatey.org/install.ps1'))"
```

Install with PowerShell.exe

With PowerShell, there is an additional step. You must ensure [Get-ExecutionPolicy](#) is not Restricted. We suggest using `Bypass` to bypass the policy to get things installed or `AllSigned` for quite a bit more security.

Run `Get-ExecutionPolicy`. If it returns `Restricted`, then run `Set-ExecutionPolicy AllSigned` or `Set-ExecutionPolicy Bypass`.

Now run the following command:  (copy command text)

```
Set-ExecutionPolicy Bypass; iex ((New-Object System.Net.WebClient).DownloadString('https://chocolatey.org/install.ps1'))
```

Additional considerations

NOTE: Please inspect <https://chocolatey.org/install.ps1> prior to running any of these scripts to ensure safety. We already know it's safe, but you should verify the security and contents of **any** script from the internet you are not familiar with. All of these scripts download a remote PowerShell script and execute it on your machine.

We take security very seriously. [Learn more](#)

OPEN CHAT

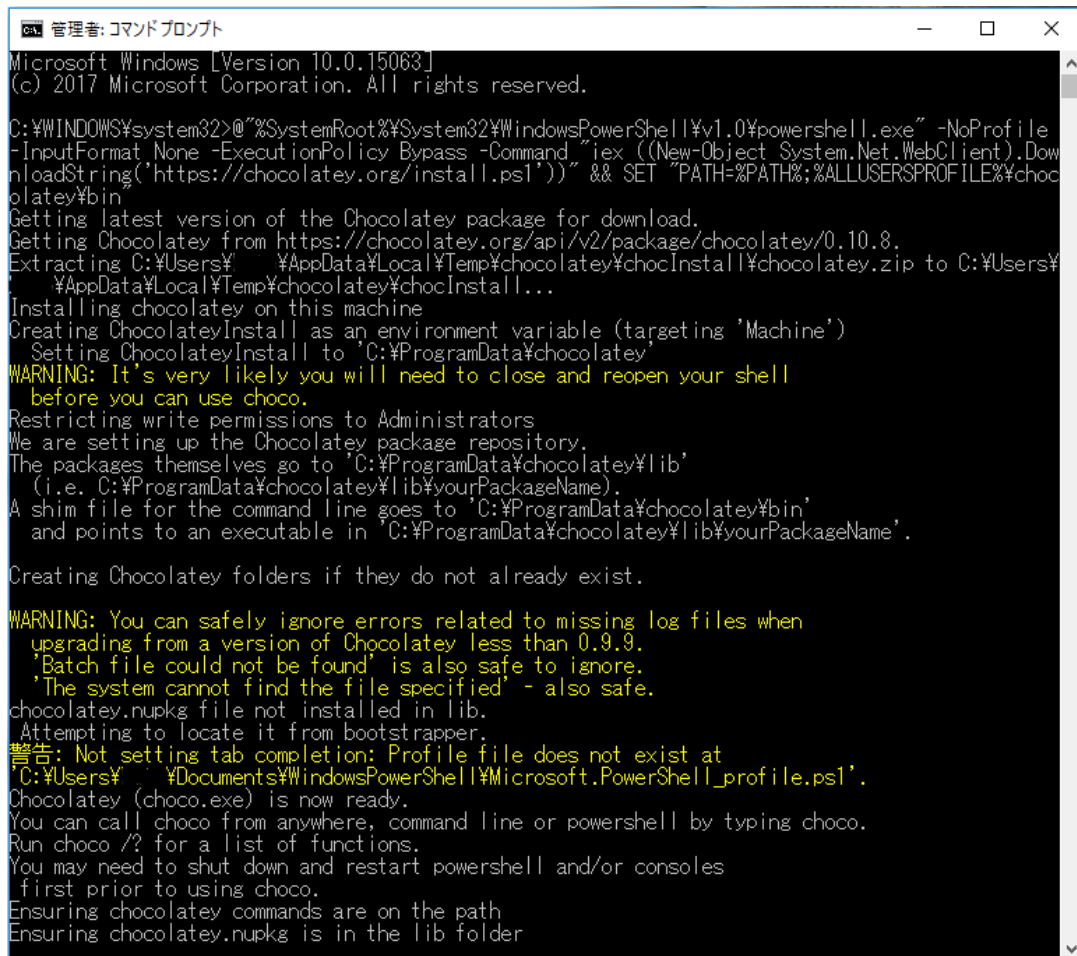
図 4.2 コマンドプロンプトでインストールする場合

赤い丸の部分を選択するとコマンドプロンプト Chocolatey をインストールするコマンドがコピーされる。

実際にコピーしたコマンドは以下である。

```
@'%SystemRoot%\System32\WindowsPowerShell\v1.0\powershell.exe' -NoProfile  
-InputFormat None -ExecutionPolicy Bypass -Command "iex ((New-Object  
System.Net.WebClient).DownloadString('https://chocolatey.org/install.ps1'))"  
&& SET "PATH=%PATH%;%ALLUSERSPROFILE%\chocolatey\bin"
```

コマンドプロンプトを管理者で起動し、先ほどコピーしたコマンドを入力すると Chocolatey をインストールする。以下の画面はインストールが成功画面である。



```
管理者: コマンドプロンプト
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>@"%SystemRoot%\System32\WindowsPowerShell\v1.0\powershell.exe" -NoProfile -InputFormat None -ExecutionPolicy Bypass -Command "iex ((New-Object System.Net.WebClient).DownloadString('https://chocolatey.org/install.ps1'))" && SET "PATH=%PATH%;%ALLUSERSPROFILE%\chocolatey\bin"
Getting latest version of the Chocolatey package for download.
Getting Chocolatey from https://chocolatey.org/api/v2/package/chocolatey/0.10.8.
Extracting C:\Users%\AppData\Local\Temp\chocolatey\chocInstall\chocolatey.zip to C:\Users%\AppData\Local\Temp\chocolatey\chocInstall...
Installing chocolatey on this machine
Creating ChocolateyInstall as an environment variable (targeting 'Machine')
Setting ChocolateyInstall to 'C:\ProgramData\chocolatey'
WARNING: It's very likely you will need to close and reopen your shell
before you can use choco.
Restricting write permissions to Administrators
We are setting up the Chocolatey package repository.
The packages themselves go to 'C:\ProgramData\chocolatey\lib'
(i.e. C:\ProgramData\chocolatey\lib\yourPackageName).
A shim file for the command line goes to 'C:\ProgramData\chocolatey\bin'
and points to an executable in 'C:\ProgramData\chocolatey\lib\yourPackageName'.

Creating Chocolatey folders if they do not already exist.

WARNING: You can safely ignore errors related to missing log files when
upgrading from a version of Chocolatey less than 0.9.9.
'Batch file could not be found' is also safe to ignore.
'The system cannot find the file specified' - also safe.
chocolatey.nupkg file not installed in lib.
Attempting to locate it from bootstrapper.
警告: Not setting tab completion: Profile file does not exist at
C:\Users%\Documents\WindowsPowerShell\Microsoft.PowerShell_profile.ps1'.
Chocolatey (choco.exe) is now ready.
You can call choco from anywhere, command line or powershell by typing choco.
Run choco /? for a list of functions.
You may need to shut down and restart powershell and/or consoles
first prior to using choco.
Ensuring chocolatey commands are on the path
Ensuring chocolatey.nupkg is in the lib folder
```

図 4.3 Chocolatey のインストール成功画面

4.2.2 R の導入

R とは

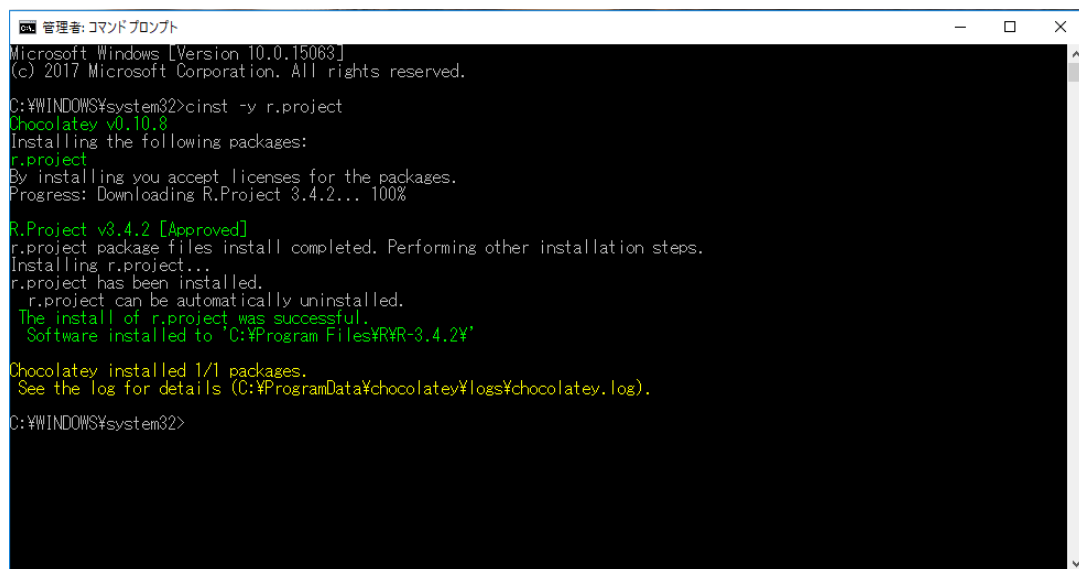
R とは統計解析ソフトであり，オープンソース・フリーソフトウェアとなっている．

Chocolatey を使った R の導入

以下のコマンドで R をインストールする．

```
choco install -y r.project
```

以下の画面はインストールが成功画面である．



```
管理: コマンドプロンプト
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>choco install -y r.project
Chocolatey v0.10.8
Installing the following packages:
r.project
By installing you accept licenses for the packages.
Progress: Downloading R.Project 3.4.2... 100%
R.Project v3.4.2 [Approved]
r.project package files install completed. Performing other installation steps.
Installing r.project...
r.project has been installed.
r.project can be automatically uninstalled.
The install of r.project was successful.
Software installed to 'C:\Program Files\R\R-3.4.2\
Chocolatey installed 1/1 packages.
See the log for details (C:\ProgramData\chocolatey\logs\chocolatey.log).

C:\WINDOWS\system32>
```

図 4.4 R のインストール

インストーラーを使った R の導入

CRAN(The Comprehensive R Archive Network) が提供する R for Windows のインストーラダウンロードページ (<https://cran.ism.ac.jp/>) にアクセスし, Download R for Windows を選択.

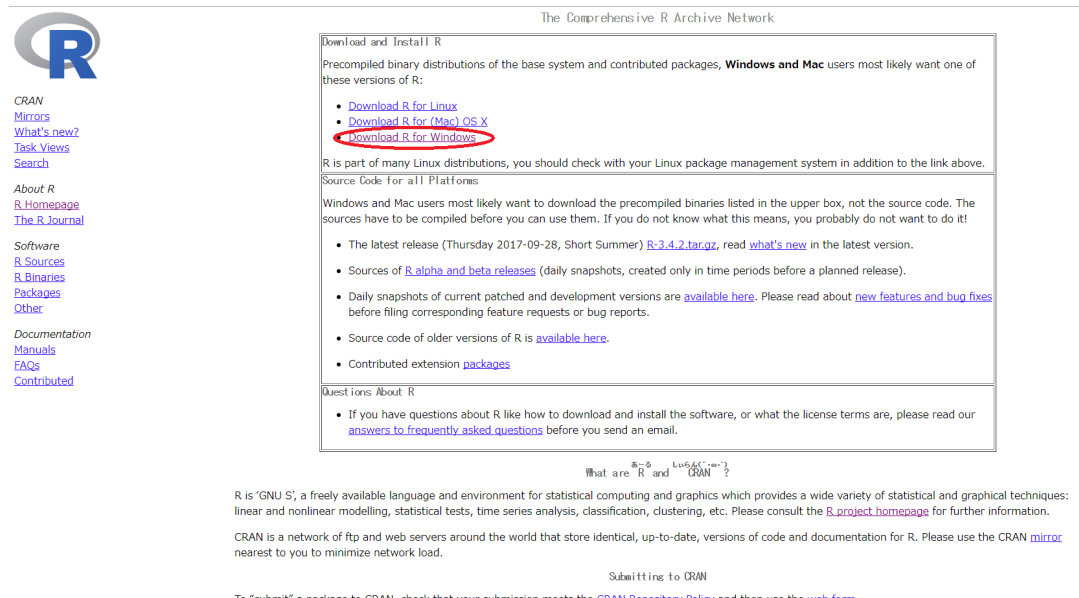


図 4.5 Download R for Windows を選択

base をインストールするので、install R for the first time を選択.

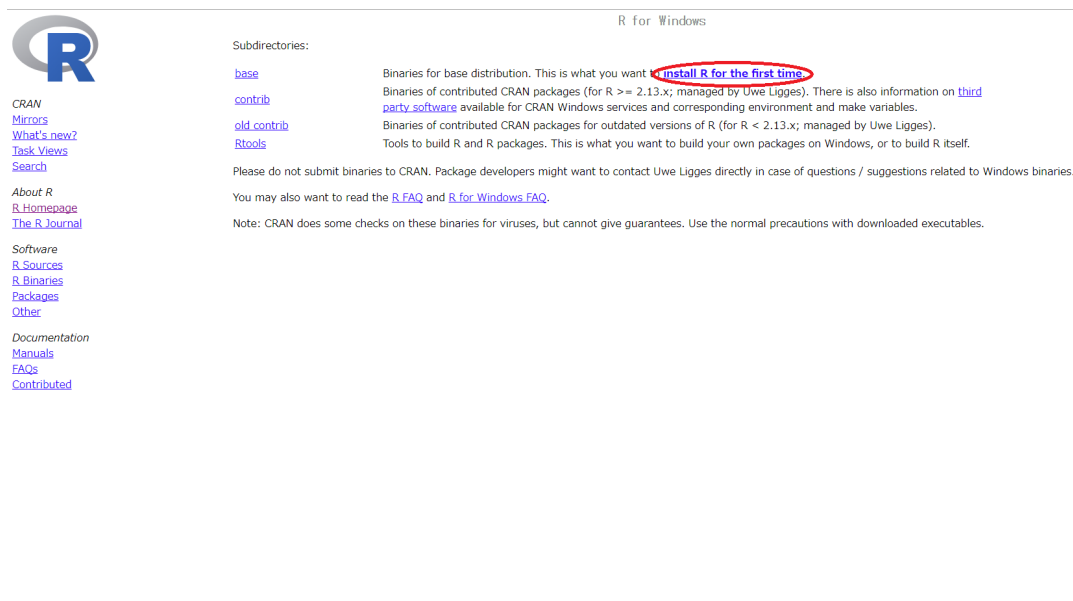
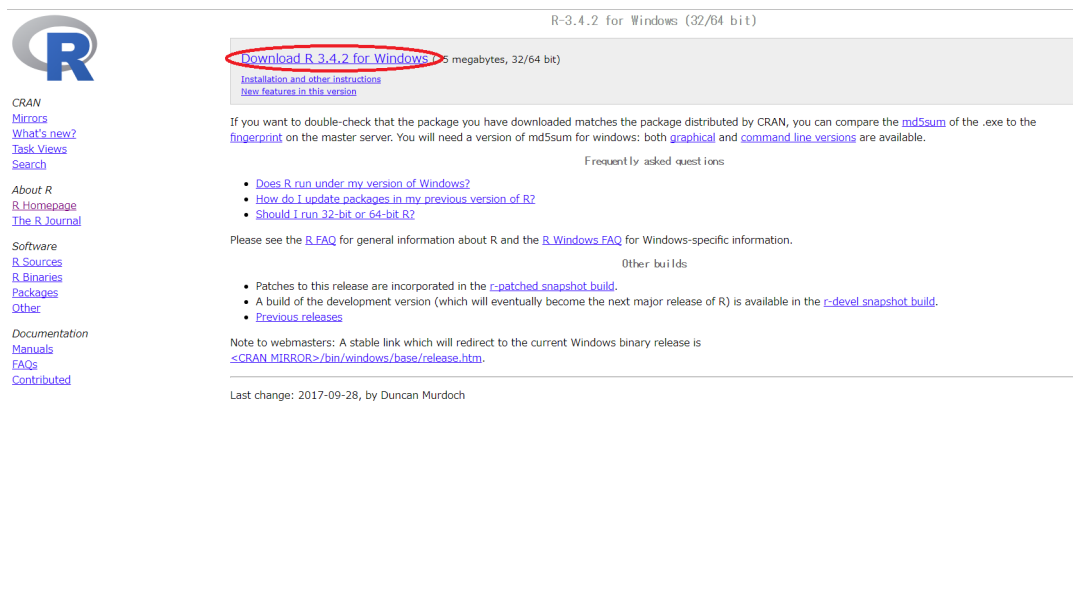


図 4.6 install R for the first time を選択

Download R 3.4.2 for Windows(3.4.2 の部分はバージョンによって異なる) を選択.



R-3.4.2 for Windows (32/64 bit)

Download R 3.4.2 for Windows

15.5 megabytes, 32/64 bit

[Installation and other instructions](#)
[New features in this version](#)

If you want to double-check that the package you have downloaded matches the package distributed by CRAN, you can compare the [md5sum](#) of the .exe to the [fingerprint](#) on the master server. You will need a version of md5sum for windows: both [graphical](#) and [command line versions](#) are available.

Frequently asked questions

- [Does R run under my version of Windows?](#)
- [How do I update packages in my previous version of R?](#)
- [Should I run 32-bit or 64-bit R?](#)

Please see the [R FAQ](#) for general information about R and the [R Windows FAQ](#) for Windows-specific information.

Other builds

- Patches to this release are incorporated in the [r-patched snapshot build](#).
- A build of the development version (which will eventually become the next major release of R) is available in the [r-devel snapshot build](#).
- [Previous releases](#)

Note to webmasters: A stable link which will redirect to the current Windows binary release is [<CRAN MIRROR>/bin/windows/base/release.htm](#).

Last change: 2017-09-28, by Duncan Murdoch

図 4.7 Download R 3.4.2 for Windows

インストーラーがダウンロードできたら起動し，インストールする．

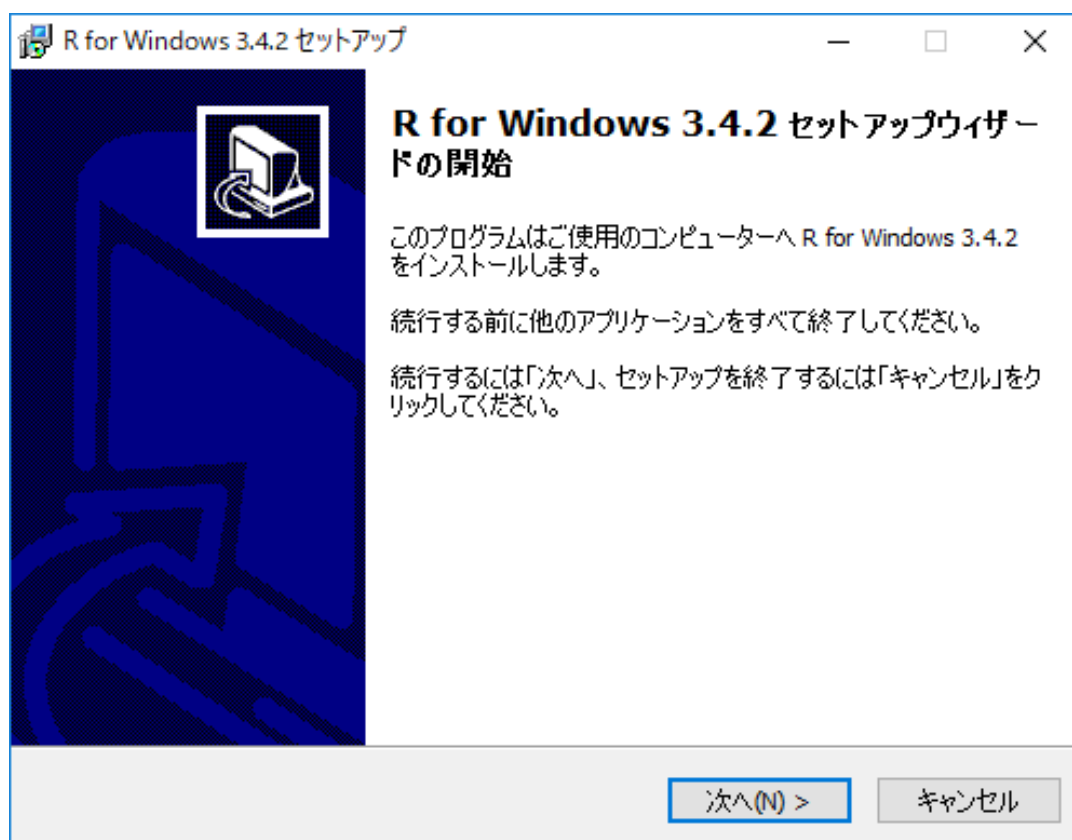


図 4.8 R をインストールする

4.2.3 Anaconda の導入

Anaconda とは

Python と Python のライブラリをセットにしたパッケージである。

Anaconda のインストール

Ubuntu の起動したコマンドプロンプト内で以下のコマンドを入力する。

```
bash Anaconda3-5.0.1-Linux-x86_64.sh
```

環境変数の設定

Ubuntu の起動したコマンドプロンプト内で以下のコマンドを入力する。

```
cat << "EOF" >> .bash_profile
export CUDA_HOME=/usr/local/cuda
export CUDA_PATH=$CUDA_HOME
export PATH=$HOME/anaconda3/bin:$CUDA_HOME/bin:$PATH
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$CUDA_HOME/lib64:$CUDA_HOME/extras/CUPTI/lib64
EOF
```


4.2.4 Twitter API の導入

Twitter API とは

Twitter 社が提供しているサービスで、Web サイトやアプリなどから Twitter の機能呼び出すことができる。この API を利用することでつぶやきの参照や検索などを行なえるアプリケーション開発ができるようになる。

tweepy のインストール

Ubuntu の起動したコマンドプロンプト内で以下のコマンドを入力する。

```
sudo pip install tweepy
```

Twitter の API キー，アクセストークンの取得

(<https://apps.twitter.com/>) にアクセスし，で新しいアプリを作る。そこで Twitter の API キー，アクセストークンを取得する。Twitter のアカウントが必要なので登録しておく必要がある。登録はメールアドレスと電話番号があれば可能である。

4.2.5 Mastodon API の導入

Mastodon API とは

Mastodon API はネット上に使用方法があげられており、Web サイトやアプリなどから Mastodon の機能呼び出すことができる。この API を利用することでつぶやきの参照や検索などを行なえるアプリケーション開発ができるようになる。

Mastodon.py のインストール

Ubuntu の起動したコマンドプロンプト内で以下のコマンドを入力する。

```
sudo pip install Mastodon.py
```

Mastodon の API キー、アクセストークンを取得

以下の Python ファイルを起動することで Mastodon の API キー、アクセストークンを取得できる。取得するためにインスタンスへ登録しておく必要がある。登録はメールアドレスがあれば可能である。

```
from mastodon import Mastodon

Mastodon.create_app(
    'pytooterapp',
    api_base_url = '*****.jp',
    to_file = 'Mastodon_clientcred.secret'
)
mastodon = Mastodon(
    client_id = 'Mastodon_clientcred.secret',
    api_base_url = '*****.jp'
)
mastodon.log_in(
    'my_login_email@example.com',
    'password',
    to_file = 'Mastodon_usercred.secret'
)
```

4.2.6 Word2vec の導入

MeCab のインストール

Ubuntu の起動したコマンドプロンプト内で以下のコマンドを入力する.

```
sudo apt install -y mecab mecab-ipadic-utf8 libmecab-dev
```

```
pip install mecab-python3
```

gensim のインストール

Ubuntu の起動したコマンドプロンプト内で以下のコマンドを入力する.

```
pip install gensim
```

word2vec 学習済みモデルのダウンロード

Ubuntu の起動したコマンドプロンプト内で以下のコマンドを入力する.

```
wget http://www.cl.ecei.tohoku.ac.jp/~m-suzuki/jawiki_vector/data/20170201.tar.bz2
```

```
tar xf 20170201.tar.bz2
```

4.3 研究に使うデータの取得

4.3.1 Twitter からつぶやきの取得

以下のファイルを `auth.py` として保存し，Twitter の API キーとアクセストークンを入力する．

```
# -*- coding: utf-8 -*-

import tweepy

consumer_key = "*****"

consumer_secret = "*****"

access_token = "*****"

access_token_secret = "*****"

auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token, access_token_secret)
api = tweepy.API(auth)
```

以下のファイルはつぶやきのランダムサンプリングを取得するコードである [3].

```
# -*- coding: utf-8 -*-

from tweepy.streaming import StreamListener
from tweepy import Stream
from auth import auth

class StdOutListener(StreamListener):
    def on_data(self, data):
        if data.startswith("{"):
            print (data)
            return True

    def on_error(self, status):
        print (status)

if __name__ == '__main__':
    stream = Stream(auth, StdOutListener())
    stream.sample()
```

以下のファイルはランダムサンプリングしたつぶやきから日本語のみを抽出するコードである [4].

```
# -*- coding: utf-8 -*-
#!/usr/bin/env python
import sys, json

for line in sys.stdin:
    try:
        tweet = json.loads(line)

        if 'retweeted_status' not in tweet:

            if tweet['user']['lang'] == 'ja':
                text = tweet['text'].encode('utf-8').replace("\n", "").replace(", ",
                    "")

                print text

    except StandardError:
        pass
```

4.3.2 Mastodon からつぶやきの取得

以下のファイルは Mastodon のインスタンスのつぶやきについている ID の 0 から 100000 まで 50 ごとに 1 つのつぶやきを取得する。

```
# -*- Coding: utf-8 -*-

from mastodon import Mastodon
from time import sleep

mastodon = Mastodon(
    client_id = 'Mastodon_clientcred.secret',
    access_token = 'Mastodon_usercred.secret',
    api_base_url = '*****.jp')

for i in range(0,100000,50):
    text = mastodon.timeline('local',max_id=20+i, since_id= -20+i, limit=1)
    print(text)
    sleep(1)
```

以下のファイルで取得したデータを扱いやすくする。

```
# -*- coding: utf-8 -*-

import sys
import datetime
from dateutil.tz import tzutc

for line in sys.stdin:
    for i in range(39):
        try:
            print('-----')
            myList = eval(line)

            if len(myList) != 0:
                dic = myList[i]
                for key in dic.keys():
                    print(key + ': ' + str(dic[key]))
        except:
            None
```


以下のコマンドでデータを抽出する.

grep content 取得したデータファイル名 > 抽出したデータファイル名

4.3.3 つぶやきのベクトル化

以下のファイルで Twitter と Mastodon の各インスタンスの 100 のつぶやきを CSV ファイルにし、ベクトル化する。

```
import sys
import gensim
import MeCab
import re
import numpy as np

model = gensim.models.KeyedVectors.load_word2vec_format('entity_vector/
    entity_vector.model.bin', binary=True)

mecab = MeCab.Tagger('-Owakati')

r = re.compile('(.*) (.*)')

def sentenceToVector(sentence):
    words = mecab.parse(sentence).strip().split()
    vec = np.zeros(model.vector_size)
    i = 0

    for word in words:
        try:
            vec = np.add(vec, model[word])
            i += 1
        except:
            pass

    if i == 0: return ve
    return np.divide(vec, i)

for line in sys.stdin:
    m = r.search(line)
    label = m.group(1)
    text = m.group(2)
    print('{0},{1}'.format(label, ','.join(map(str, list(sentenceToVector(text)
        )))))
```

4.4 つぶやきの主成分分析

4.4.1 主成分分析とは

相関のある多数の変数から相関のない少数で全体のばらつきを最もよく表す主成分と呼ばれる変数を合成する多変量解析の一つ。

4.4.2 ggbiplot のインストール

```
install.packages('devtools')  
devtools::install_github('vqv/ggbiplot')
```

4.4.3 主成分分析で使ったコード

```
setwd('c:/vagrant/machine') //ディレクトリの設定

myData <- read.csv('mstdn.csv', head = F) //myData に mstdn.csv の読み込みコマンドを代入

myResult <- prcomp(myData[, -1]) //myResult に主成分分析の結果を代入

(myLabels = 1:length(levels(myData[, 1])))

library(ggbiplot)

ggbiplot(myResult, var.axes = F, groups = myData[, 1])
```

第 5 章

結果

Twitter と 30 の Mastodon のインスタンスを対象に調査した。Twitter と機械学習の相談ができるインスタンスである `aidon.club` の 100 のつぶやきをベクトル化し、主成分分析をした結果である。

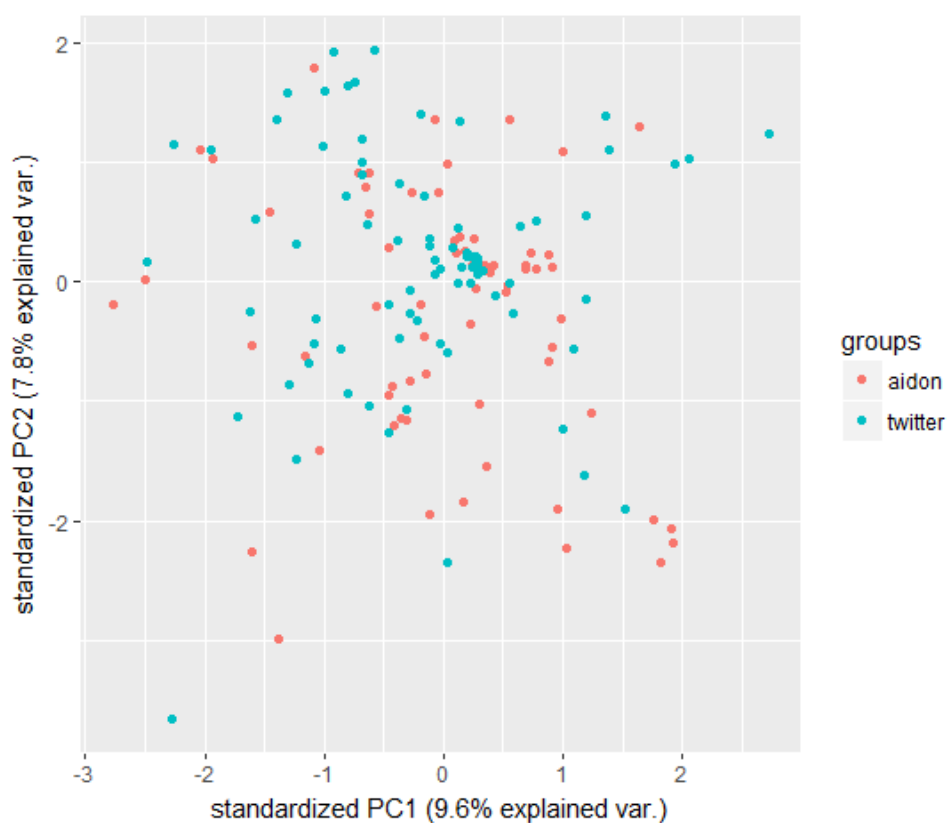


図 5.1 機械学習の相談ができるインスタンス

Twitter とスポーツバイクの話題が中心のインスタンスである `bicyclemstdn.jp` の 100 の つぶやきをベクトル化し、主成分分析をした結果である。

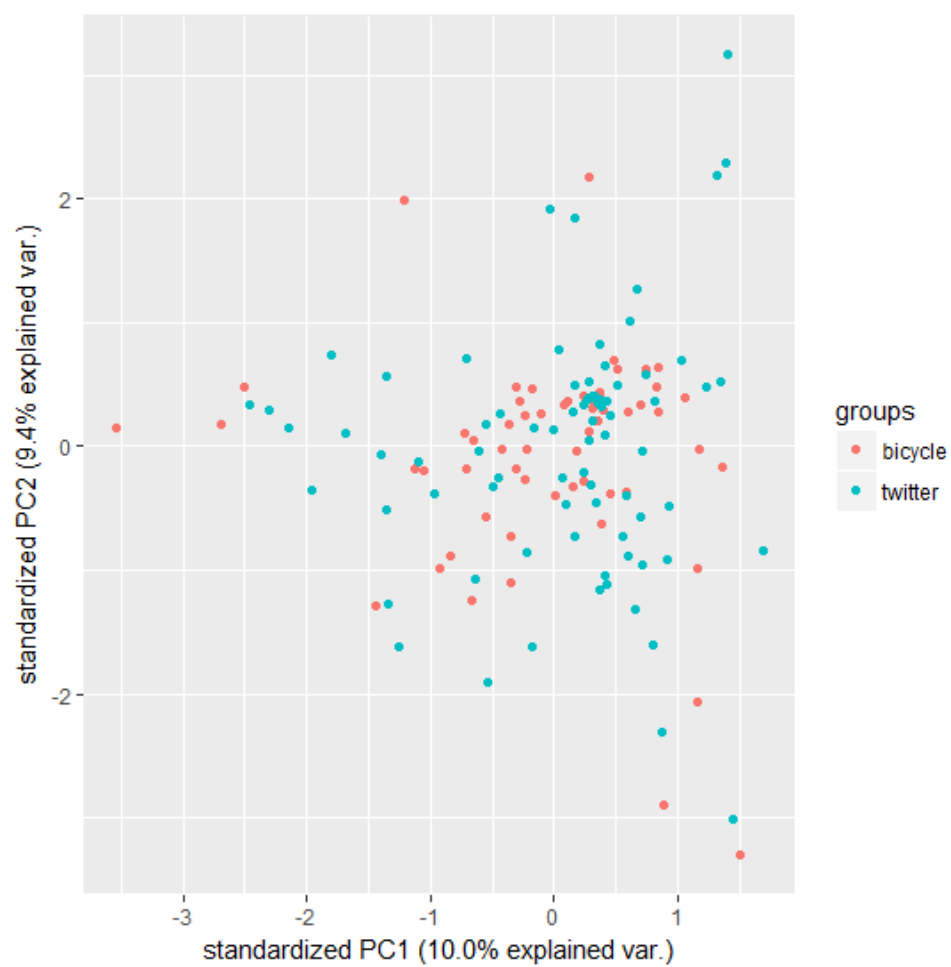


図 5.2 スポーツバイクの話題が中心のインスタンス

Twitter と猫好きのためのインスタンスである catdon.life の 100 のつぶやきをベクトル化し、主成分分析をした結果である。

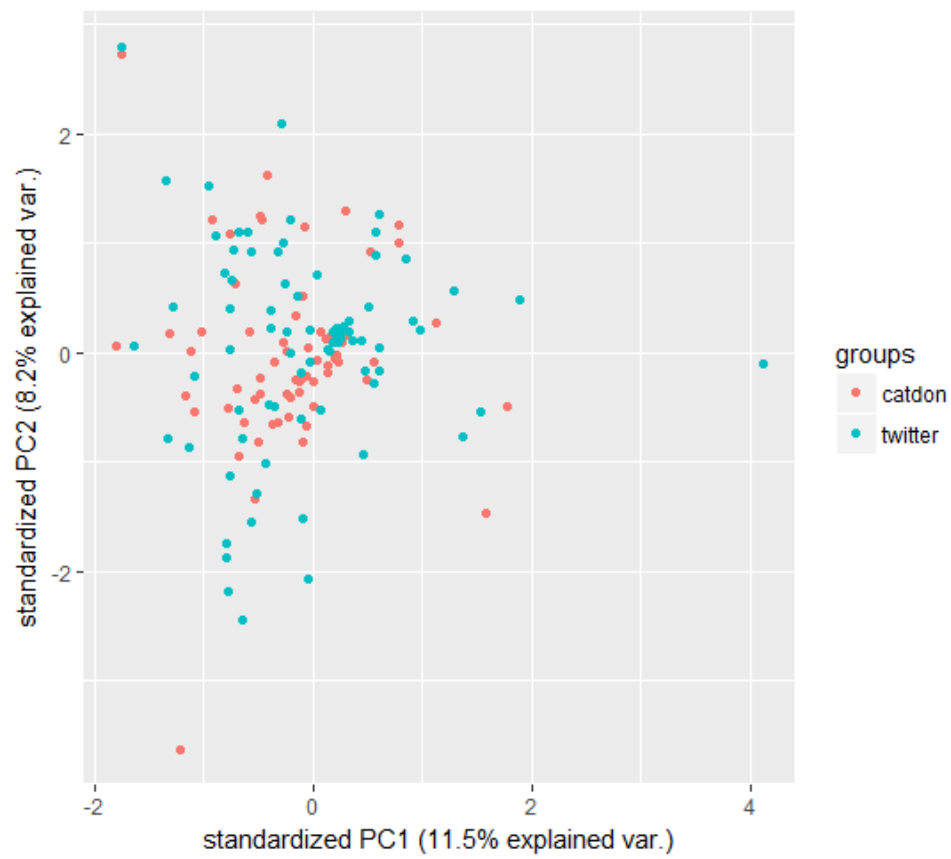


図 5.3 猫好きのためのインスタンス

Twitter とドラゴンクエスト 10 の話題が中心のインスタンスである dq10.online の 100 のつぶやきをベクトル化し、主成分分析をした結果である。

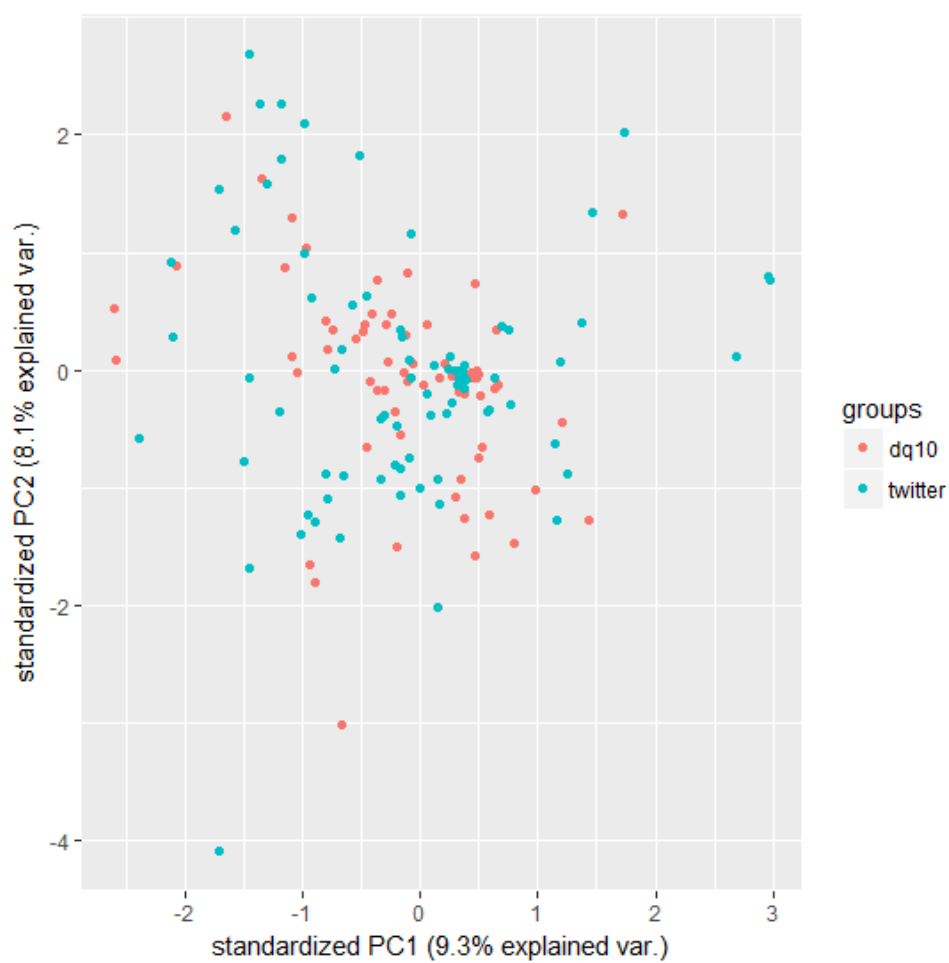


図 5.4 ドラゴンクエスト 10 の話題が中心のインスタンス

Twitter と映画好きのためのインスタンスである eigadon.net の 100 のつぶやきをベクトル化し、主成分分析をした結果である。

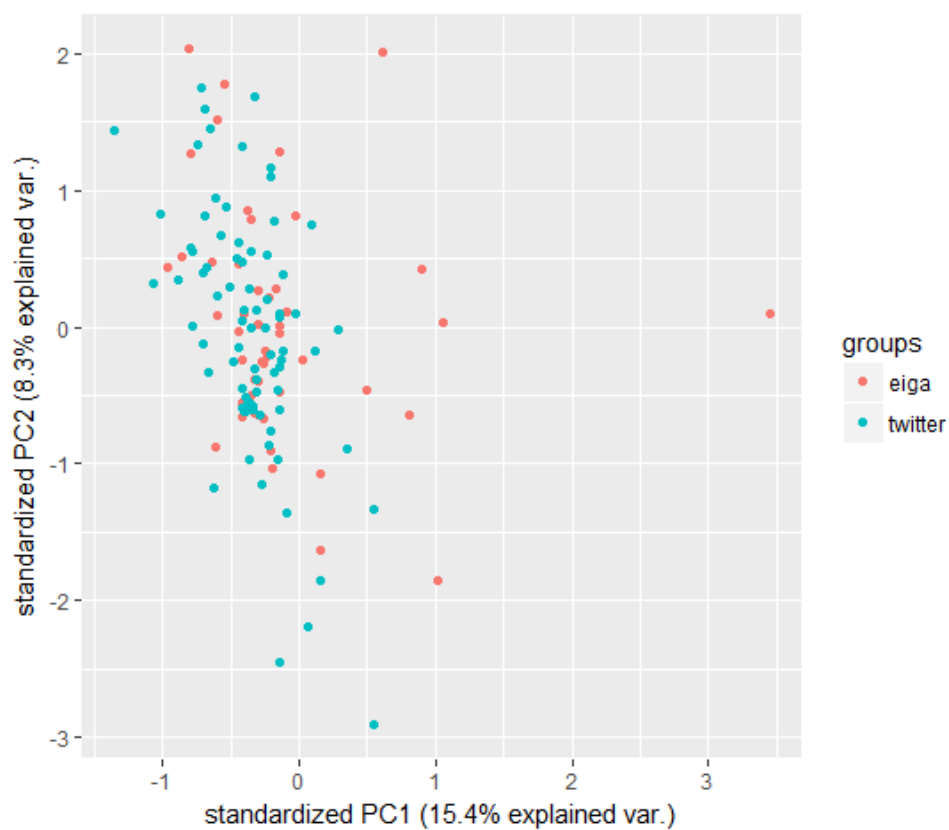


図 5.5 映画好きのためのインスタンス

Twitter と型月作品の話題が中心のインスタンスである fgochiho.vip の 100 のつぶやきをベクトル化し、主成分分析をした結果である。

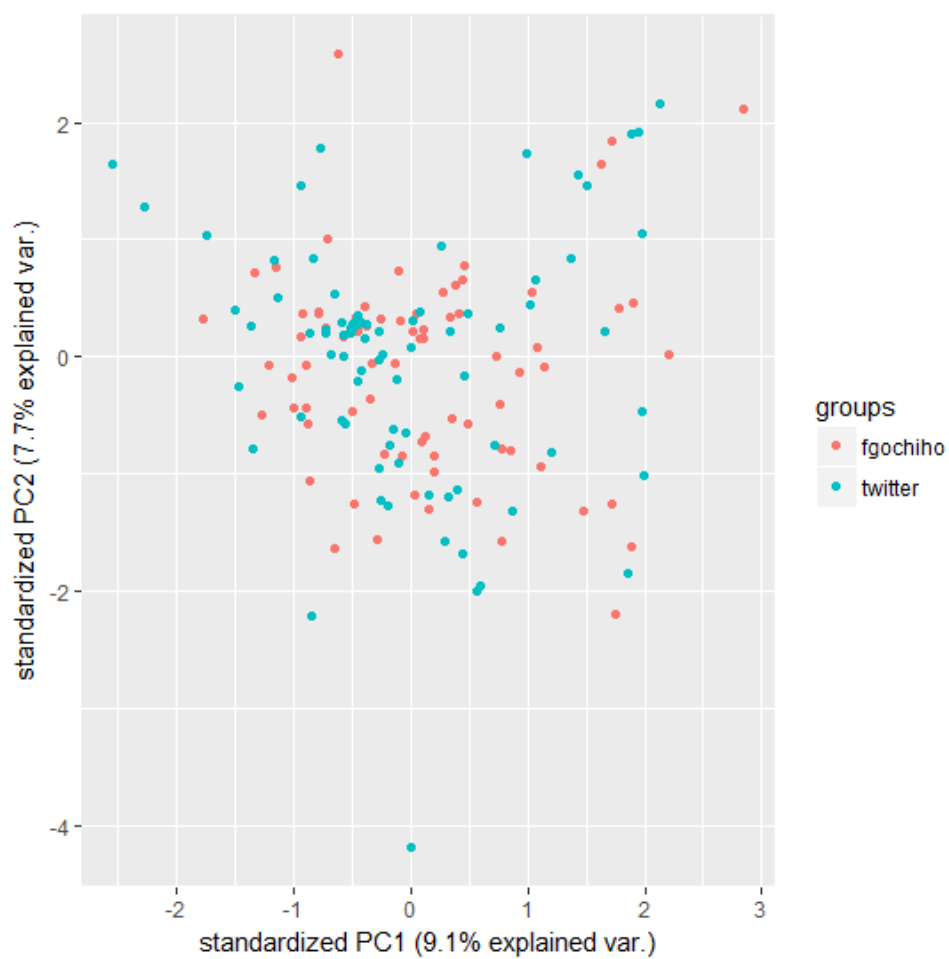


図 5.6 型月作品の話題が中心のインスタンス

Twitter とドワンゴが運営するインスタンスである friends.nico の 100 のつぶやきをベクトル化し、主成分分析をした結果である。

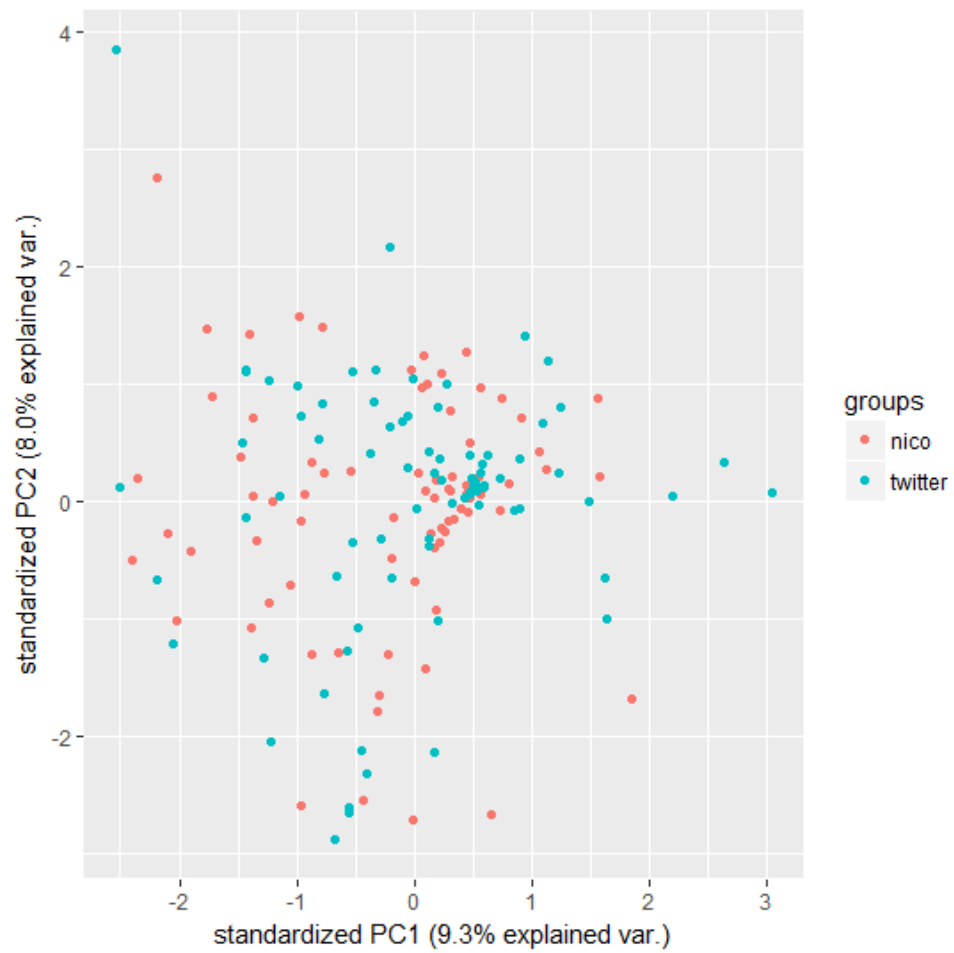


図 5.7 ドワンゴが運営するインスタンス

Twitter とゲーム制作者のインスタンスである gamecreate.mstdn.cloud の 100 のつぶやきをベクトル化し、主成分分析をした結果である。



図 5.8 ゲーム制作者のインスタンス

Twitter と Splatoon の話題が中心のインスタンスである ika.quecloud.net の 100 のつぶやきをベクトル化し、主成分分析した結果である。

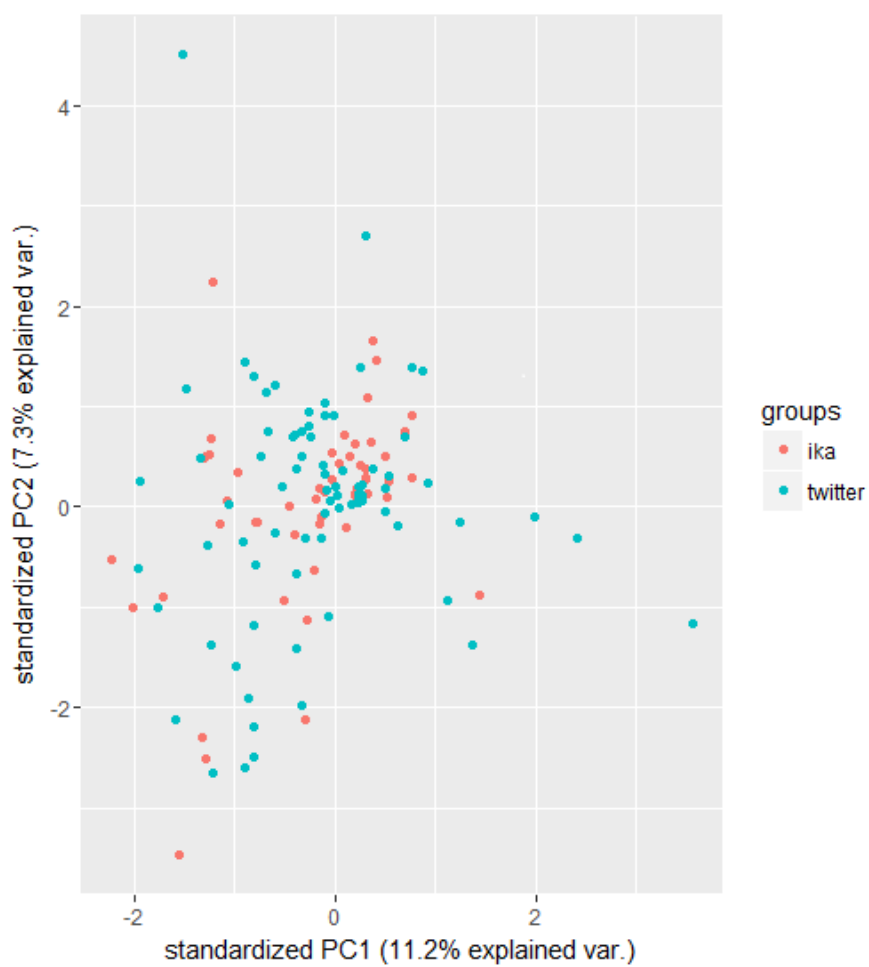


図 5.9 Splatoon の話題が中心のインスタンス

Twitter とアイドルマスターの話題が中心のインスタンスである imastodon.net の 100 の つぶやきをベクトル化し、主成分分析した結果である。

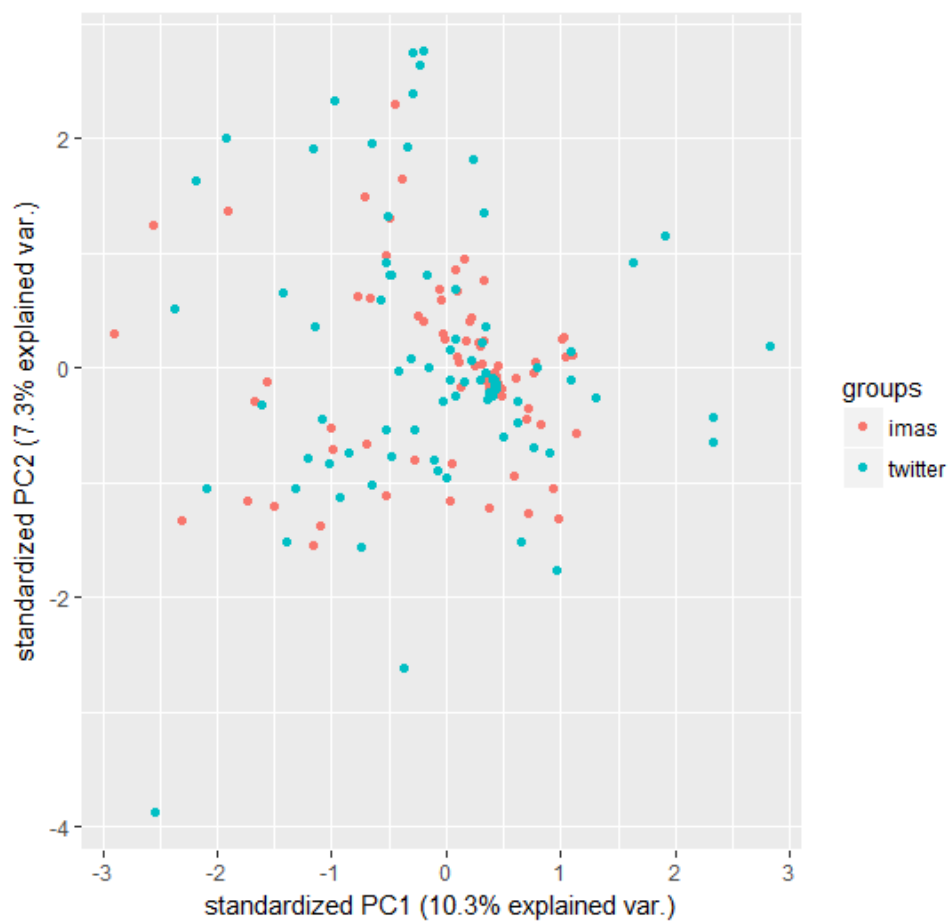


図 5.10 アイドルマスターの話題が中心のインスタンス

Twitter とカードキャプターさくら/CLAMP の話題が中心のインスタンスである kero.ccsakura.jp の 100 のつぶやきをベクトル化し、主成分分析した結果である。

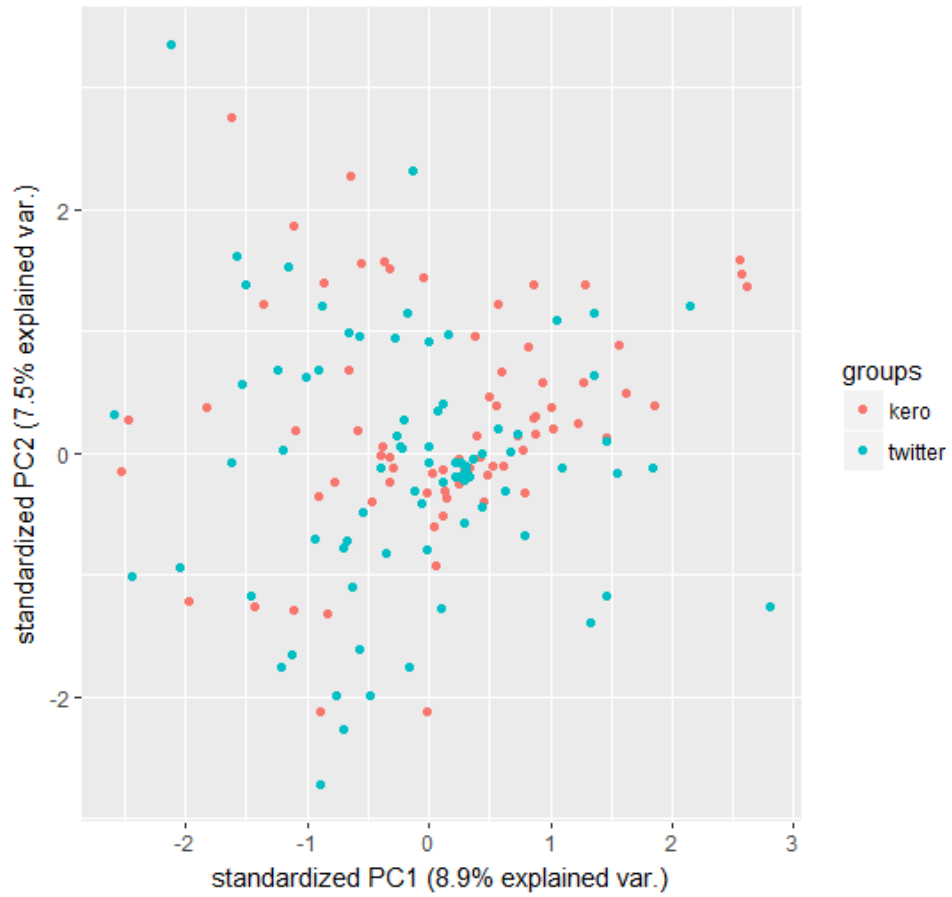


図 5.11 カードキャプターさくら/CLAMP の話題が中心のインスタンス

Twitter とアイカツ！ の話題が中心のインスタンスである kirakiratter.com の 100 のつぶやきをベクトル化し、主成分分析した結果である。

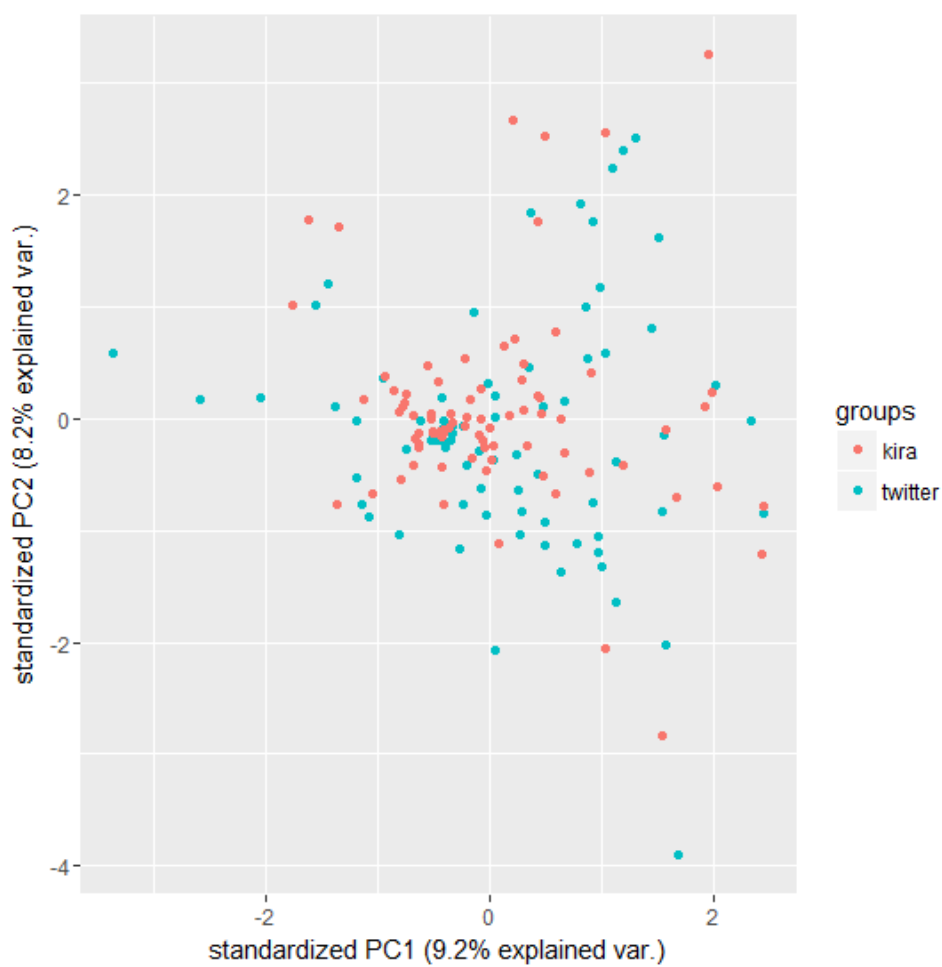


図 5.12 アイカツ！ の話題が中心のインスタンス

Twitter と婚活している人が集うインスタンスである konkat.jp の 100 のつぶやきをベクトル化し、主成分分析した結果である。

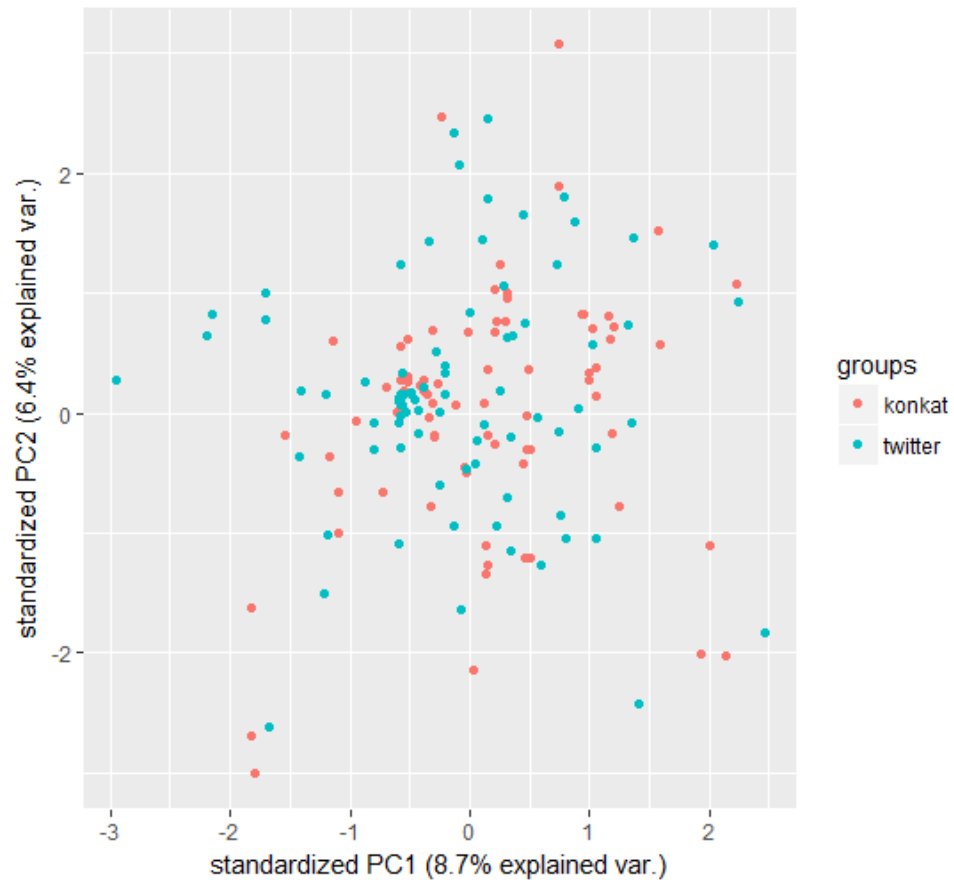


図 5.13 婚活している人が集うインスタンス

Twitter とクラゲ専門のインスタンスである kurage.cc の 100 のつぶやきをベクトル化し、主成分分析した結果である。

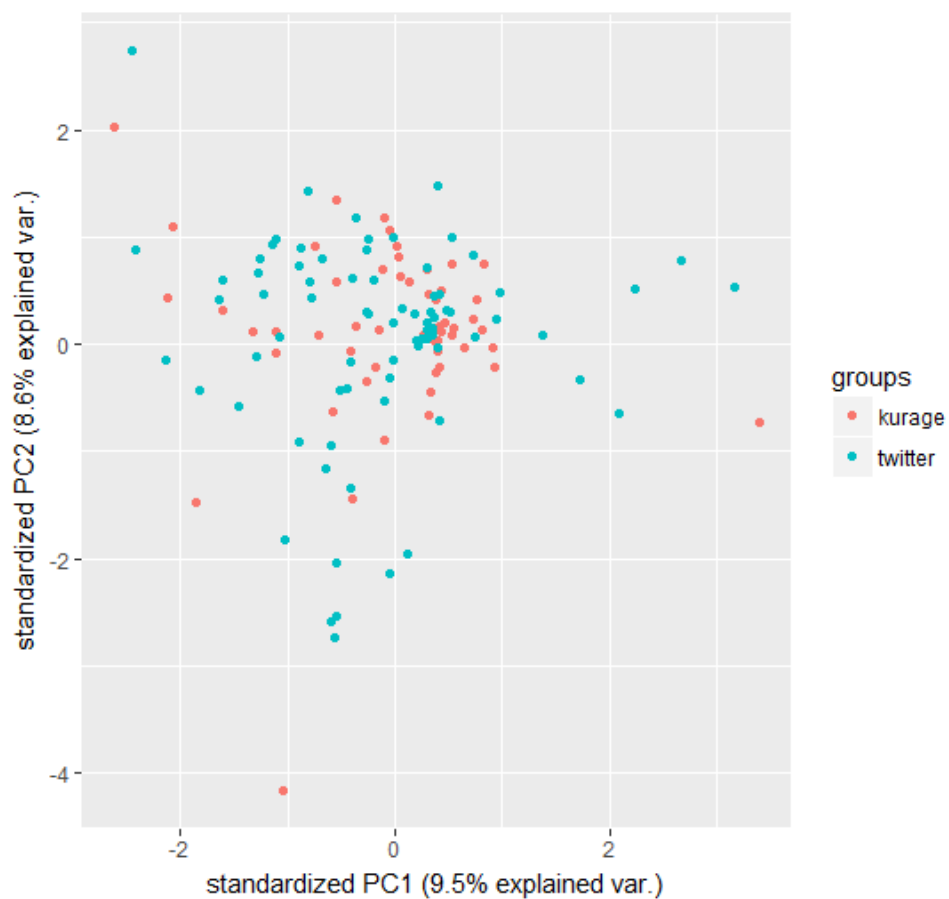


図 5.14 クラゲ専門のインスタンス

Twitter とアニメ，ゲームの話題が中心のインスタンスである mast.moe の 100 のつぶやきをベクトル化し，主成分分析した結果である．



図 5.15 アニメ，ゲームの話題が中心のインスタンス

Twitter と仮想通貨ユーザーのためのインスタンスである mastodon.bitbank.cc の 100 の つぶやきをベクトル化し、主成分分析した結果である。

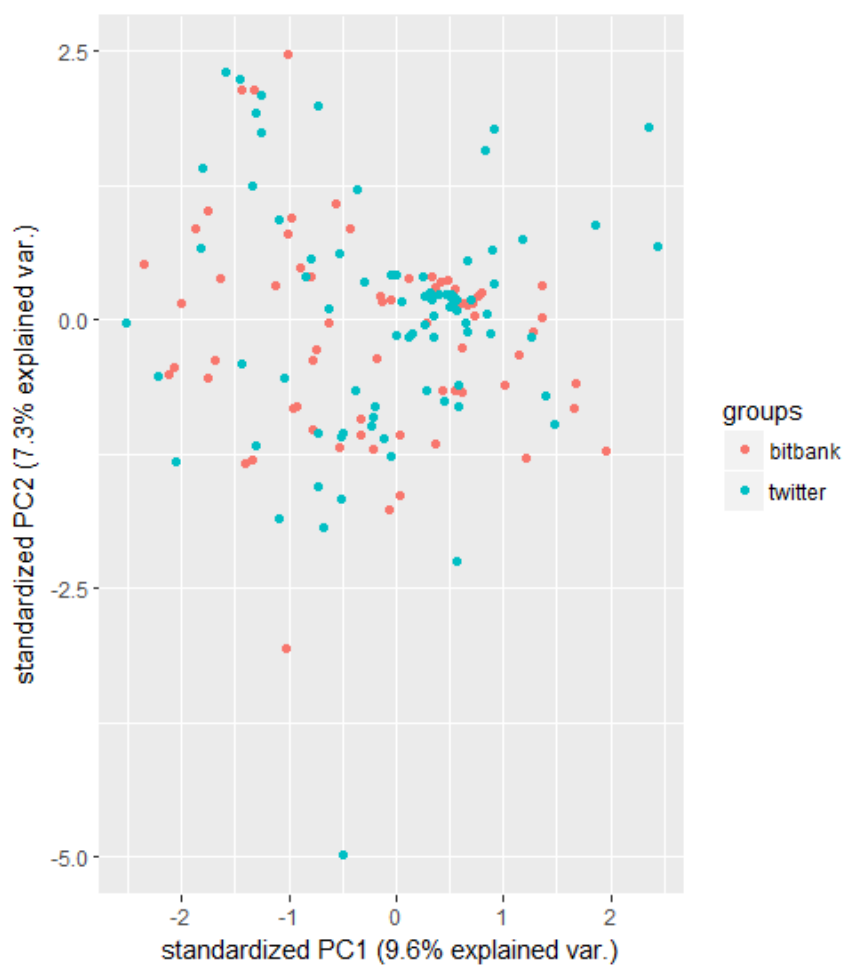


図 5.16 仮想通貨ユーザーのためのインスタンス

Twitter と宇宙開発と天文観測の話題が中心のインスタンスである [mastodon.cosmicanimal.jp](https://mstdn.social/@cosmicanimal) の 100 のつぶやきをベクトル化し、主成分分析した結果である。

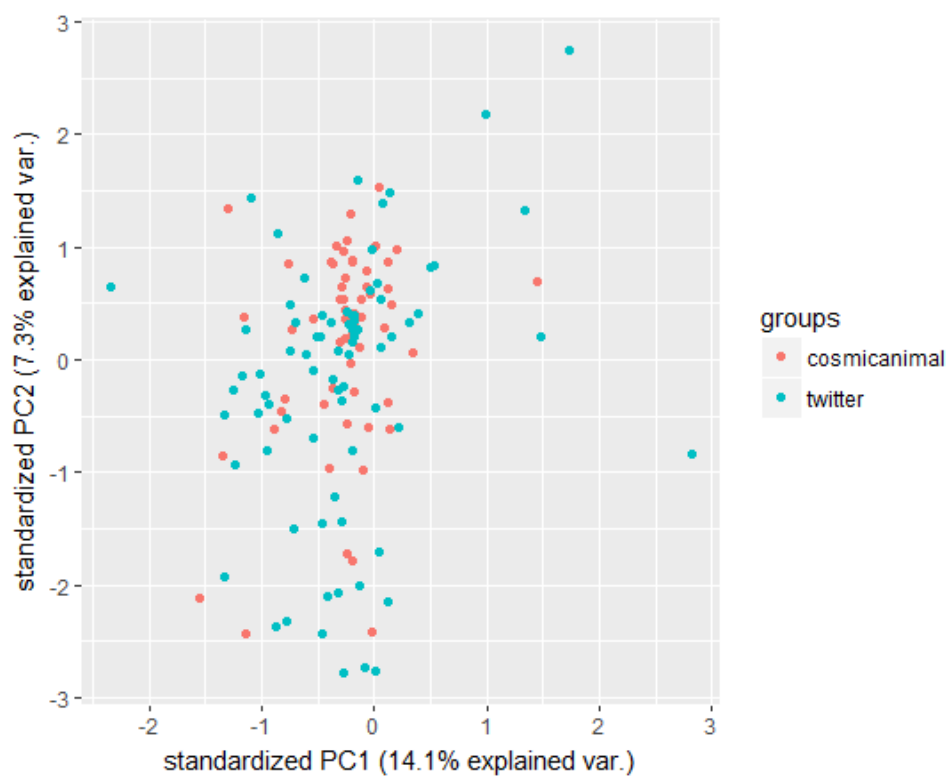


図 5.17 宇宙開発と天文観測の話題が中心のインスタンス

Twitter と釣り人専用のインスタンスである mastodon.fishing の 100 のつぶやきをベクトル化し、主成分分析した結果である。

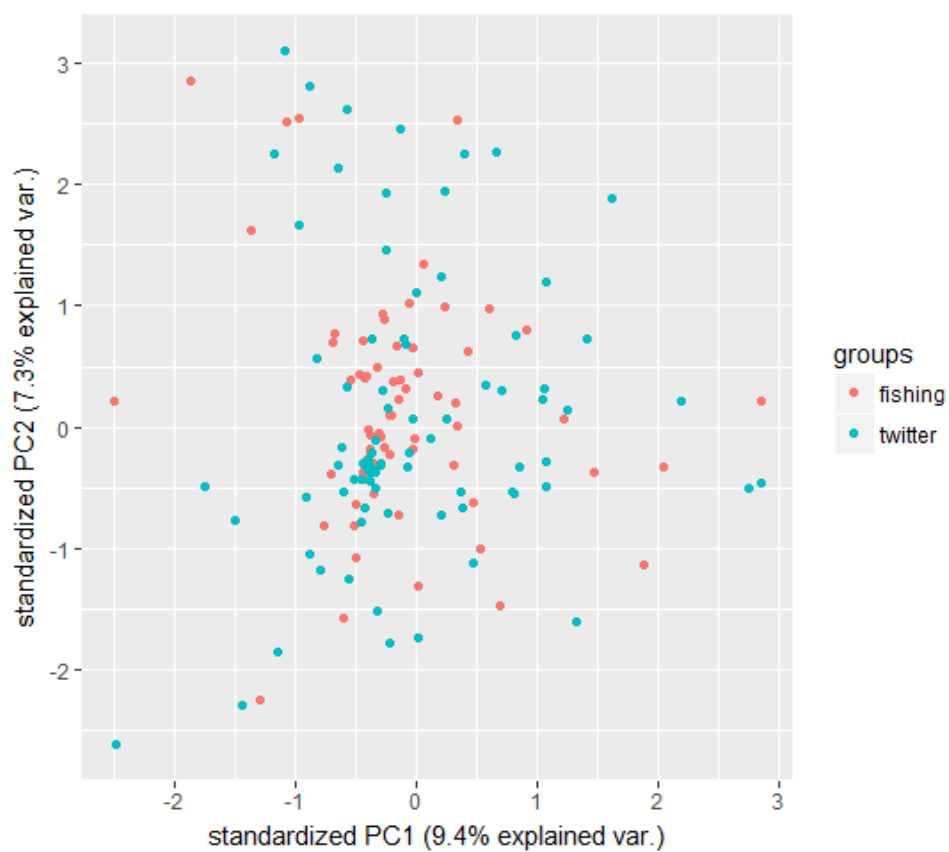


図 5.18 釣り人専用のインスタンス

Twitter と横浜の話題が中心のインスタンスである mastodon.yokohama の 100 のつぶやきをベクトル化し，主成分分析した結果である．

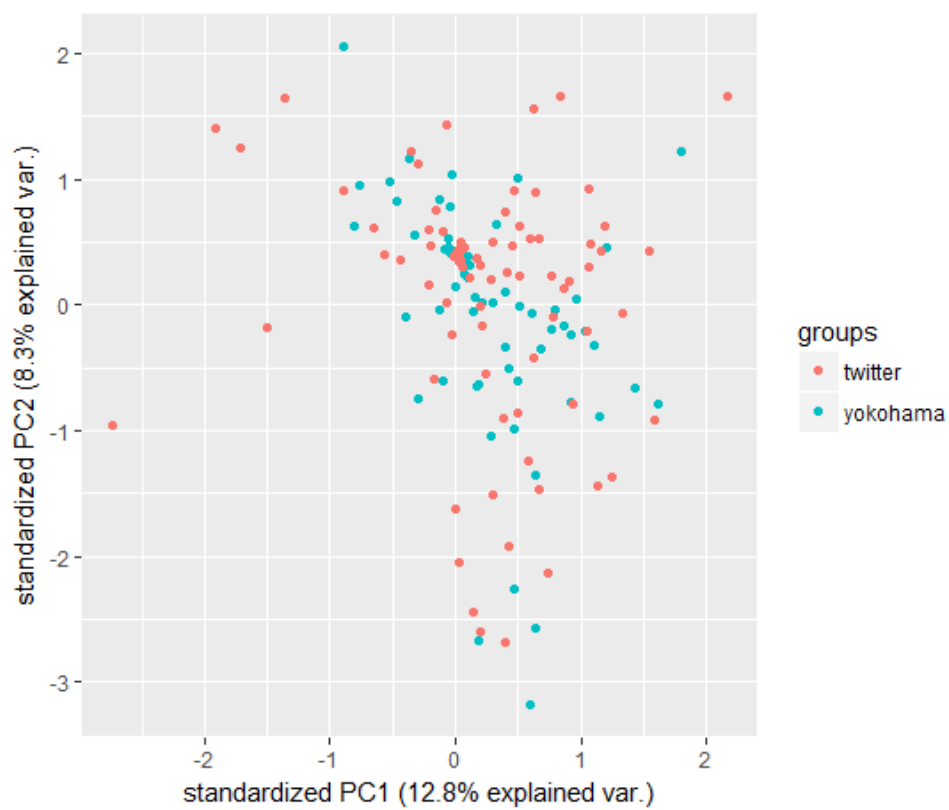


図 5.19 横浜の話題が中心のインスタンス

Twitter と北海道の話題が中心のインスタンスである mstdn.hokkaido.jp の 100 のつぶやきをベクトル化し，主成分分析した結果である．



図 5.20 北海道の話題が中心のインスタンス

Twitter と話題が自由なインスタンスである mstdn.jp の 100 のつぶやきをベクトル化し、主成分分析をした結果である。

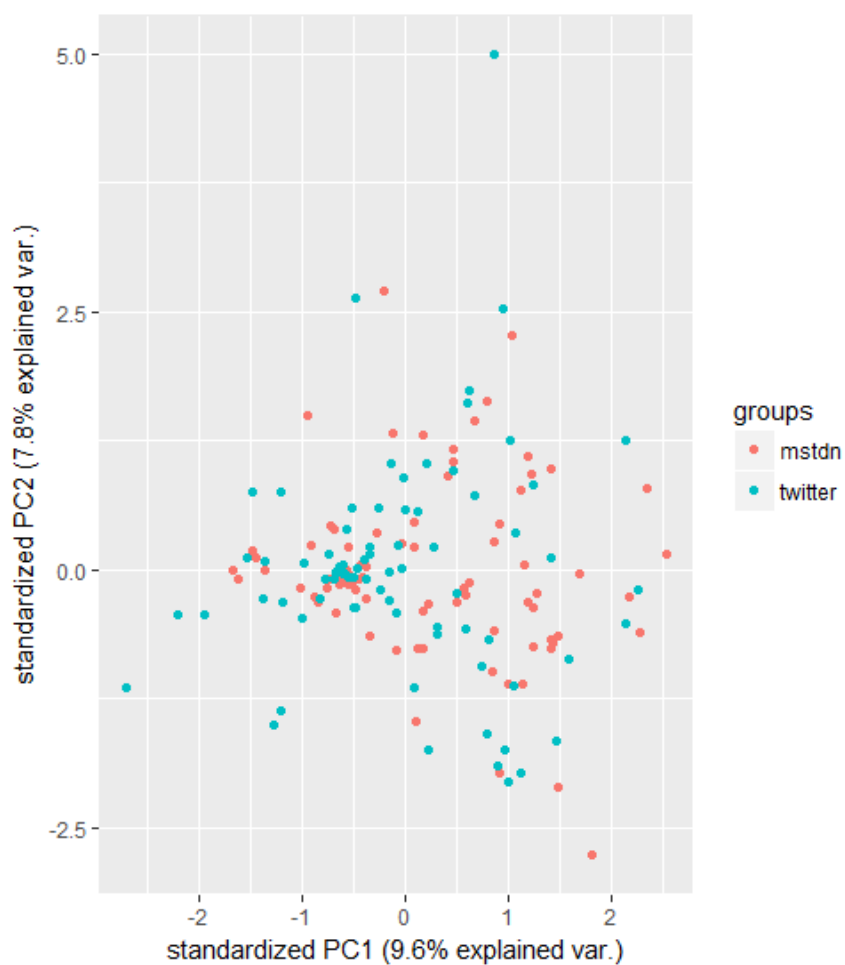


図 5.21 話題が自由なインスタンス

Twitter と大阪の話題が中心のインスタンスである mstdn.osaka の 100 のつぶやきをベクトル化し、主成分分析をした結果である。



図 5.22 大阪の話題が中心のインスタンス

Twitter とサッカーの話題が中心のインスタンスである mstdn-football.jp の 100 のつぶやきをベクトル化し、主成分分析をした結果である。

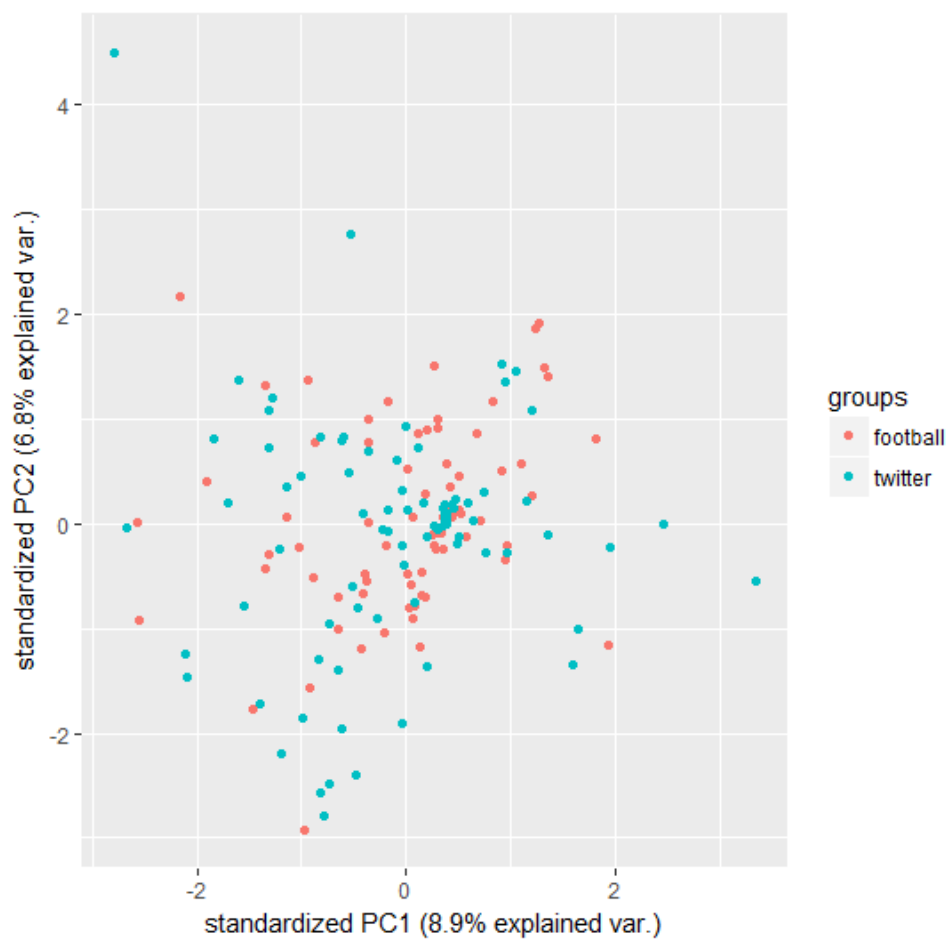


図 5.23 サッカーの話題が中心のインスタンス

Twitter と金沢市の話題が中心のインスタンスである mstdn-kanazawa.jp の 100 のつぶやきをベクトル化し，主成分分析をした結果である．

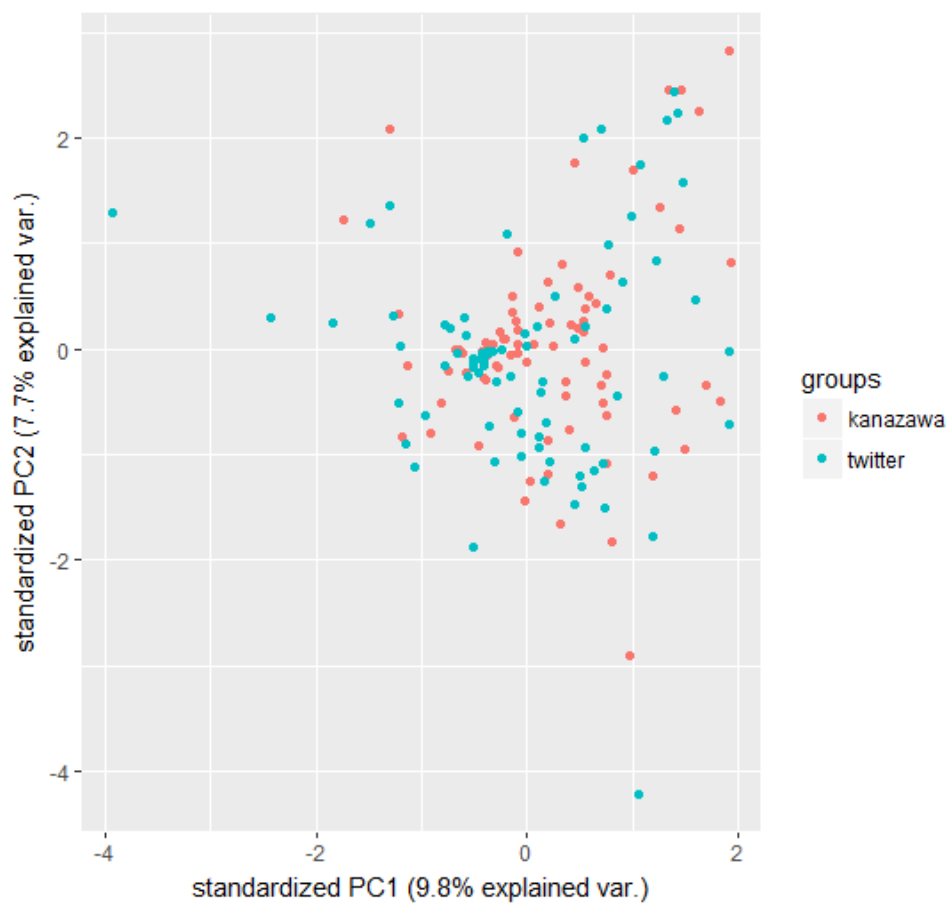


図 5.24 金沢市の話題が中心のインスタンス

Twitter ときぼうソフトが運営する、初めて Facebook ログインと BBCode による文字の回転に対応したインスタンスである now.kibousoft.co.jp の 100 のつぶやきをベクトル化し、主成分分析をした結果である。

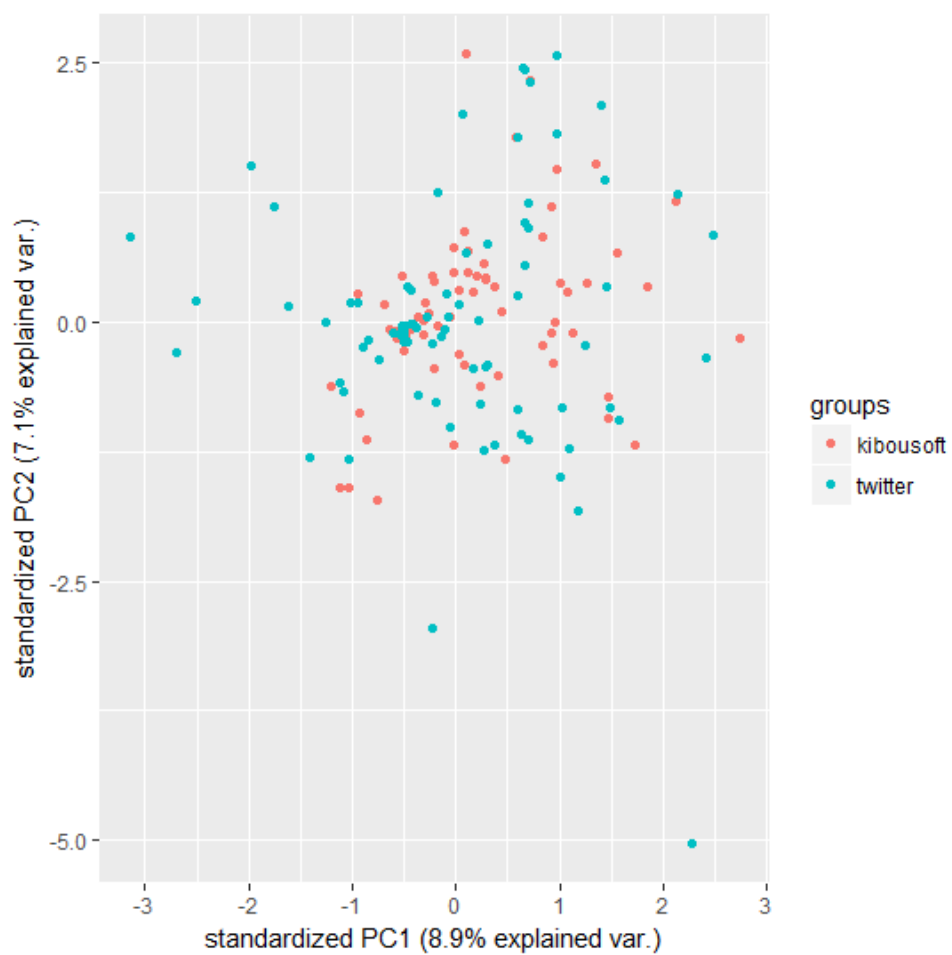


図 5.25 きぼうソフトが運営するインスタンス

Twitter とピクシブが運営するインスタンスである pawoo.net の 100 のつぶやきをベクトル化し、主成分分析をした結果である。

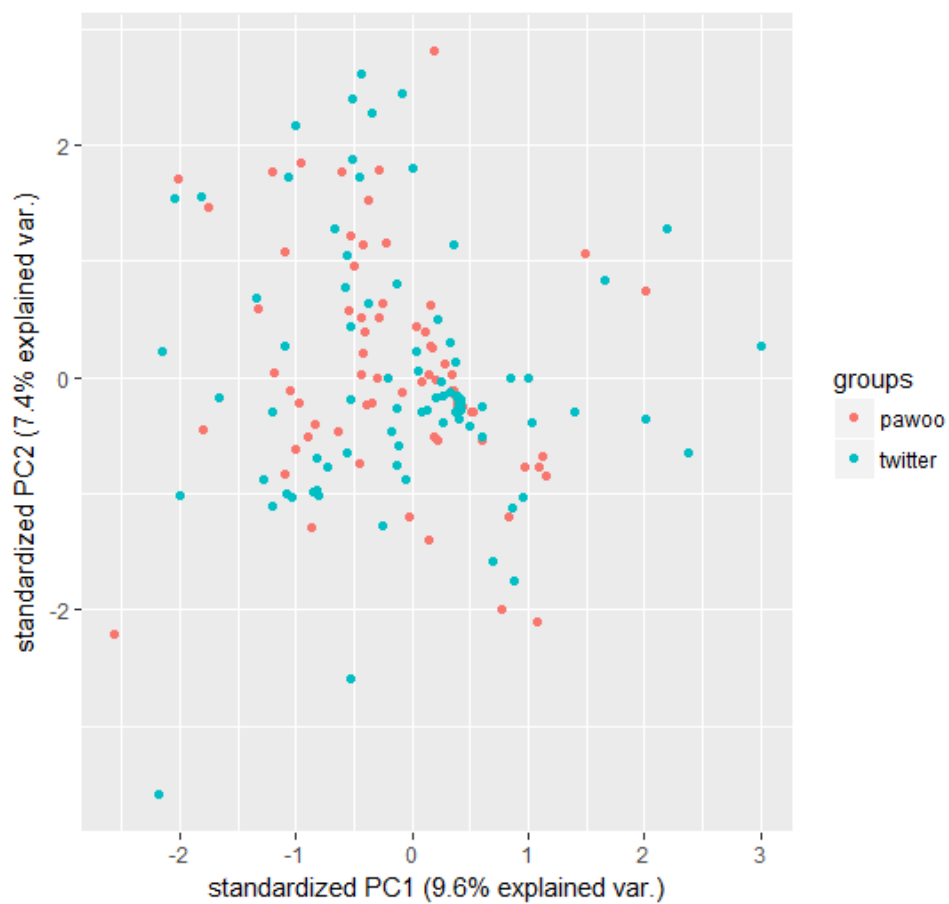


図 5.26 ピクシブが運営するインスタンス

Twitter とラグナロクオンラインの話題が中心のインスタンスである ro-mastodon.puyo.jp の 100 のつぶやきをベクトル化し、主成分分析をした結果である。

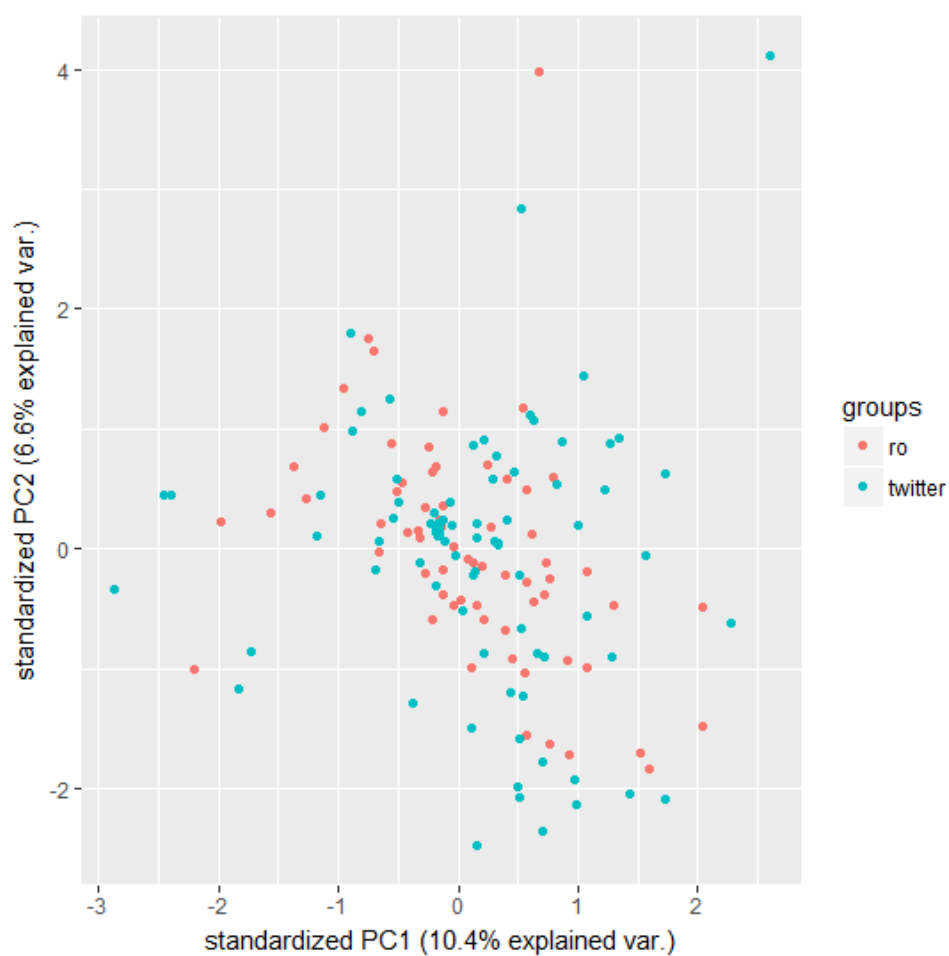


図 5.27 ラグナロクオンラインの話題が中心のインスタンス

Twitter とオープンソースのプロジェクト管理ツール「Redmine」の話題が中心のインスタンスである ro-mastodon.puyo.jp の 100 のつぶやきをベクトル化し、主成分分析をした結果である。

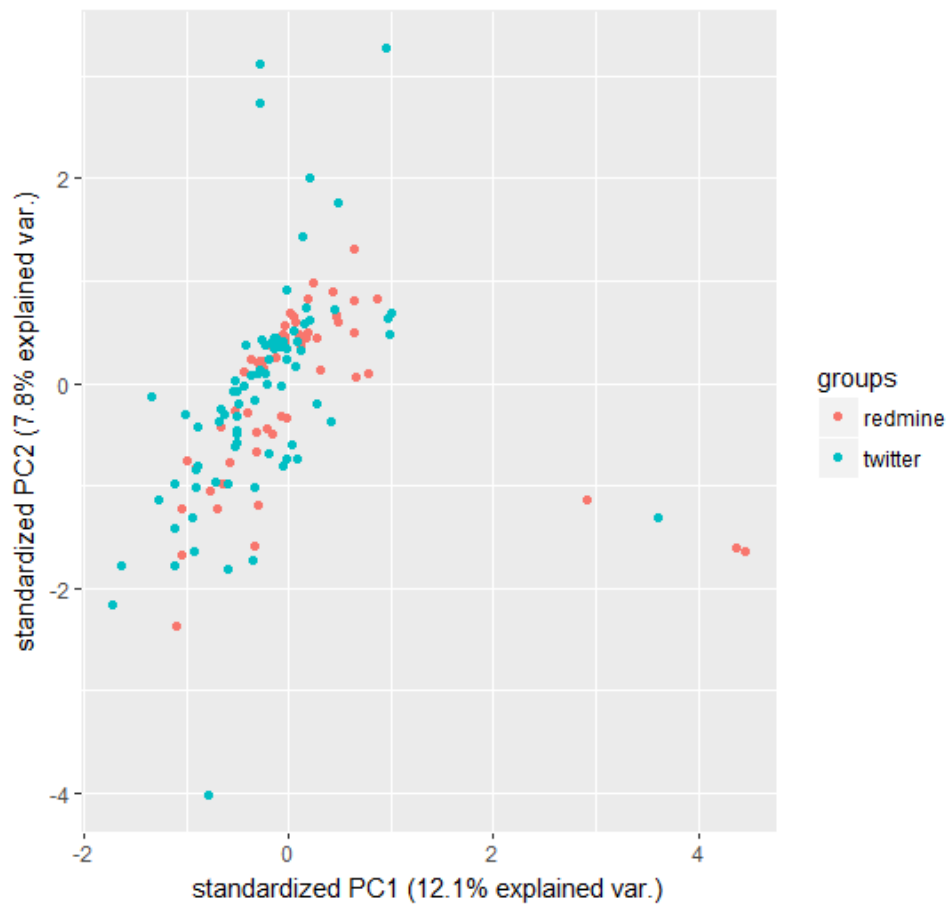


図 5.28 オープンソースのプロジェクト管理ツール「Redmine」の話題が中心のインスタンス

Twitter とニッポン放送が運営するインスタンスである tuner.1242.com の 100 のつぶやきをベクトル化し、主成分分析をした結果である。

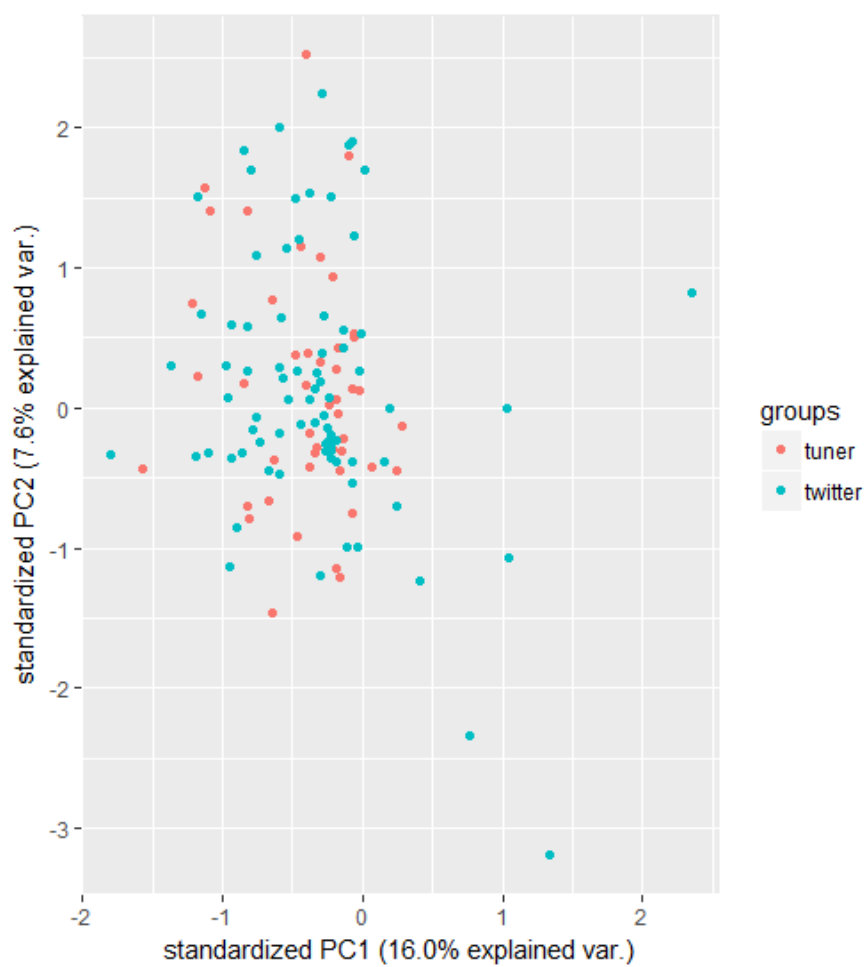


図 5.29 ニッポン放送が運営するインスタンス

Twitter とボカロクラスタが集うインスタンスである vocalodon.net の 100 のつぶやきをベクトル化し、主成分分析をした結果である。

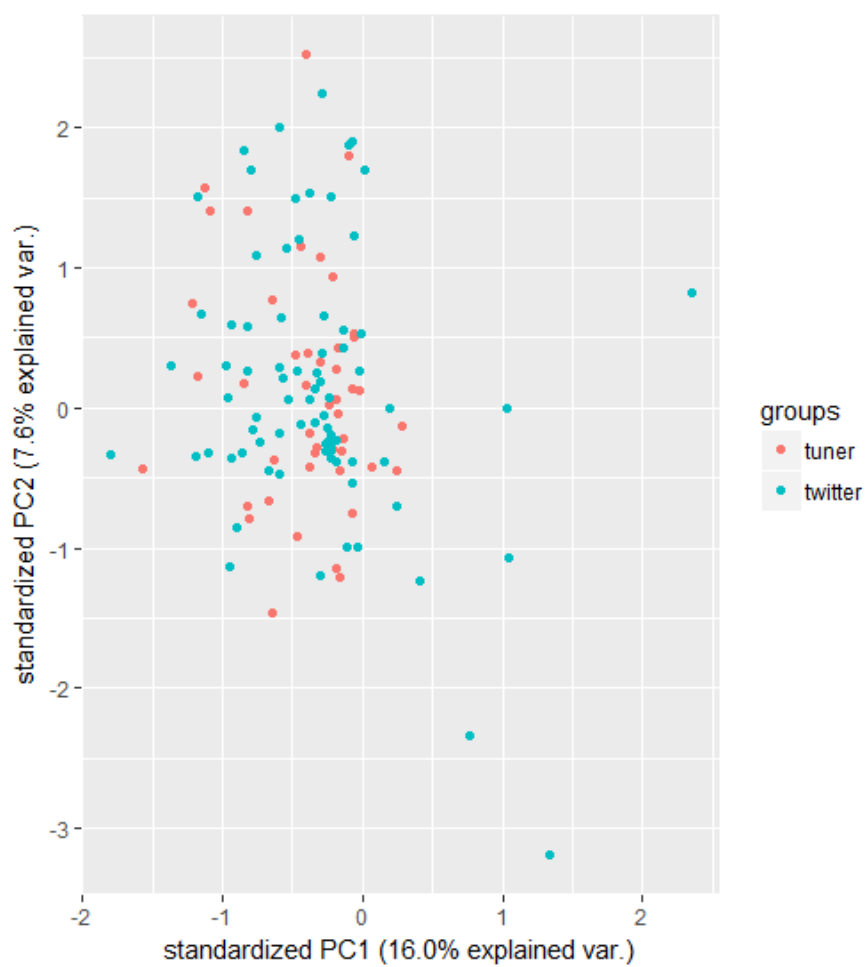


図 5.30 ボカロクラスタが集うインスタンス

第 6 章

考察

主成分分析の結果を可視化したバイプロットでは、話題が幅広い Twitter のつぶやきは拡散し、話題が限定されている Mastodon のつぶやきは局所化することが予想されたのだが、分析結果は図のように、両者に明確な違いは見られなかった。このことは、Word2vec と主成分分析という方法では、人間が簡単に理解しているような、話題の違いを検出できないことを示唆している。

第 7 章

結論

本研究で用いた Word2vec と主成分分析という手法で話題の広さの違いを識別することは困難だということが分かった．つぶやき単体ではなく，大量のつぶやきをまとめてベクトル化する手法を試みるのが今後の課題であろう．

参考文献

- [1] 武者良太. ツイッターはもう古い！？ 仲間内で楽しむSNSが人気. 日経 PC21, 2017.
- [2] 小林啓倫, コグレマサト, いしたにまさき, まつもとあつし, 堀正岳. マストドン 次世代ソーシャルメディアのすべて. 株式会社マイナビ出版, 2017.
- [3] 矢吹太朗. Streaming api で大量のつぶやきをリアルタイムに保存する方法 (python 編). <http://blog.unfindable.net/archives/4257> (2018.01.20 閲覧).
- [4] 矢吹太朗. Streaming api で取得したつぶやきの処理方法. <http://blog.unfindable.net/archives/4302> (2018.01.20 閲覧).

謝辞

本論文を作成するにあたり，ご指導を頂いた卒業論文指導教員の矢吹太郎准教授に感謝致します．先生にはお話を通じて社会を見る視野を広げていただきました．そして多くのご指摘やご助言を下さいました矢吹研究室の皆様に心から感謝します．