



総務省統計研究研修所



独立行政法人

統計センター

2020.12.19. データ解析環境Rの整備と利用

e-Stat APIを用いたスクレイピング

—家計調査のクラスタリング分析例—

和田 かず美
木村 敦

総務省統計研究研修所
独立行政法人統計センター

本日の内容

- I. e-Stat アプリケーションIDの取得
- II. 取得データの特定
- III. Rによるスクレイピング
- IV. 分析1. 都道府県庁所在市の食料消費
- V. 分析2. 東西日本を分ける食料品目
- VI. まとめ

I. e-Stat アプリケーションIDの取得



スクレイピングの作業手順

1. e-Stat [<https://www.e-stat.go.jp/>] にユーザ登録する
2. e-Statにログインして「マイページ」を表示し「API機能（アプリケーションID発行）」を選択
3. 表示される画面に、ID（任意の名前をつける）とURL（公開サイトで利用しない場合は「http://test.localhost/」等）、及び概要（何に使うか）を書き込むと、appId欄に固有のアプリケーションID（英数字の羅列）が表示される
4. 入手したい統計表のIDを調べ、必要に応じてデータの絞り込み条件を設定してRからスクレイピングする

e-Stat のトップページ

<https://www.e-stat.go.jp/>



統計で見る日本

e-Statは、日本の統計が閲覧できる政府統計ポータルサイトです

[お問い合わせ](#) | [ヘルプ](#) | [English](#)

[ログイン](#)

[新規登録](#)

[統計データを探す](#) [統計データの活用](#) [統計データの高度利用](#) [統計関連情報](#) [リンク集](#)

●統計データを探す (政府統計の調査結果を探します)

[その他の絞り込み](#)

 **すべて**

政府統計一覧の中から探します

 **分野**

17の統計分野から探します

 **組織**

統計を作成した府省等から探します

キーワード検索:

[検 索](#)

●統計データを活用する

 **グラフ**

主要指標をグラフで表示
(統計ダッシュボード)

 **時系列表**

主要指標を時系列表で表示
(統計ダッシュボード)

 **地図**

地図上に統計データを表示
(統計GIS)

 **地域**

都道府県、市区町村の
主要データを表示

 **利用ガイド**

●統計データの高度利用

マイクロデータの利用

公的統計のマイクロデータの利用案内

開発者向け

API、LODで統計データを取得

●統計関連情報

統計分類・調査項目

統計分類、市区町村コード、調査項目を表示

ログイン後の画面



政府統計の総合窓口

統計で見る日本

e-Statは、日本の統計が閲覧できる政府統計ポータルサイトです

お問い合わせ | ヘルプ | English



統計データを探す | 統計データの活用 | 統計データの高度利用 | 統計関連情報 | リンク集

登録メールアドレス | **マイページ** | ログアウト

● 統計データを探す

(政府統計の調査結果を探します)



すべて

政府統計一覧の中から探します



分野

17の統計分野から探します



組織

統計を作成した府省等から探します

その他の絞り込み

キーワード検索 :

検索

● 統計データを活用する



グラフ

主要指標をグラフで表示
(統計ダッシュボード)



時系列表

主要指標を時系列表で表示
(統計ダッシュボード)



地図

地図上に統計データを表示
(統計GIS)



地域

都道府県、市区町村の
主要データを表示

利用ガイド

● 統計データの高度利用

マイクロデータの利用
公的統計のマイクロデータの利用案内

開発者向け
API、LODで統計データを取得

● 統計関連情報

統計分類・調査項目
統計分類、市区町村コード、調査項目を表示



新着情報

一覧表示

2020-12-16

林野庁

特用林産物生産統計調査

ファイル



公表予定

一覧表示

2020-12-18 08:30

総務省

小売物価統計調査【動向編】主要品目の都市別小売価格
(都道府県庁所在市及び人口15万以上の市)(2020年11月分)



ランキング

(検索キーワード) | 一覧表示

月次

1

人口8672

2

国勢調査6996

6

マイページ

e-Stat
政府統計の総合窓口

統計で見る日本

e-Statは、日本の統計が閲覧できる政府統計ポータルサイトです

[お問い合わせ](#) | [ヘルプ](#) | [English](#)

[統計データを探す](#) [統計データの活用](#) [統計データの高度利用](#) [統計関連情報](#) [リンク集](#)

[登録メールアドレス](#)

[マイページ](#) [ログアウト](#)

[トップページ](#) / [ダッシュボード](#)

[マイページ登録](#)

[ユーザ情報変更](#)

[メールによる配信](#)

[API機能\(アプリケーションID発行\)](#)

[ダッシュボード](#)


ダッシュボード

 **統計データ（データベース）**

[すべて表示](#)

政府統計名

- 民間企業の勤務条件制度等調査 
- 国家公務員死因調査 
- 国家公務員災害補償統計 
- 退職公務員生活状況調査 
- 一般職の国家公務員の任用状況調査 

 **統計データ（ファイル）**

[すべて表示](#)

政府統計名

- 民間企業退職金実態調査
- 一般職国家公務員在職状況統計表（人事統計報告）
- 国家公務員退職手当実態調査
- 民間企業の勤務条件制度等調査 
- 国家公務員死因調査 

アプリケーションID発行

マイページ表示

統計データ

新着情報

公表予定

レイアウト一覧

統計絞り込み条件

検索履歴

ダウンロード履歴

API機能

アプリケーションIDの取得

説明

1	*名称	分析用データDL	発行 変更 廃止
	*URL	http://test.localhost/ ※http://localhostやプライベートIPアドレス（127.0.0.1等）は登録できません	
	概要	e-Statからのデータのダウンロード用	
	appId		

必要事項を記入して「発行」ボタンを押すと、appIdの欄にスクレイピングに必要な英数字の羅列が表示される

URL等の書き方



政府統計の総合窓口

統計で見る日本

e-Statは、日本の統計が閲覧できる政府統計ポータルサイトです

お問い合わせ | ヘルプ | English

統計データを探す

トップページ /

マイページ表

統計データ

アプリケーションIDの取得

説明

API機能で使用するアプリケーションIDを作成します。表示された文字列を、appIdパラメータに設定してください。

登録可能なアプリケーションIDは、3つまでとなります。

登録するアプリケーションIDごとに下記情報を入力し、発行してください。各種情報は後で変更しても構いません。

名称：API機能を利用するアプリケーション名（システム名等）

URL：アプリケーションのURL（トップページ等のURL）

※公開サイトで利用しない場合は「http://test.localhost/」等を入力してください

概要：アプリケーション（システム）の概要

*は必須入力です。

1	*名称	分析用データDL	発行 変更 廃止
	*URL	http://test.localhost/ ※http://localhostやプライベートIPアドレス（127.0.0.1等）は登録できません	
	概要	e-Statからのデータのダウンロード用	
	appId		

利用ガイド・マニュアル①

e-Statのトップページのメニューバーから「統計データの高度利用」⇒「開発者向け」⇒「API」を選択、さらに表示される次の画面で「API」を選択すると、API機能のホームページが表示される

e-Stat
政府統計の総合窓口

統計で見る日本
e-Statは、日本の統計が閲覧できる政府統計ポータルサイトです

お問い合わせ | ヘルプ | English

統計データを探す 統計データの活用 **統計データの高度利用** 統計関連情報 リンク集

マイページ ログアウト

マイクロデータの利用

●統計データを探す

開発者向け **API** 他の絞込

LOD

すべて
政府統計一覧の中から探します

分野
17の統計分野から探します

組織
統計を作成した府省等から探します

キーワード検索： 例：国勢調査 検索

●統計データを活用する

グラフ
主要指標をグラフで表示
(統計ダッシュボード)

時系列表
主要指標を時系列表で表示
(統計ダッシュボード)

地図
地図上に統計データを表示
(統計GIS)

地域
都道府県、市区町村の
主要データを表示

利用ガイド

●統計データの高度利用

マイクロデータの利用
公的統計のマイクロデータの利用案内

開発者向け
API、LODで統計データを取得

●統計関連情報

統計分類・調査項目
統計分類、市区町村コード、調査項目を表示

利用ガイド・マニュアル②

e-Statのトップページのメニューバーから「統計データの高度利用」⇒「開発者向け」⇒「API」を選択、さらに表示される次の画面で「API」を選択すると、API機能のホームページが表示される

e-Stat
政府統計の総合窓口

統計で見る日本
e-Statは、日本の統計が閲覧できる政府統計ポータルサイトです

お問い合わせ | ヘルプ | English

統計データを探す 統計データの活用 **統計データの高度利用** 統計関連情報 リンク集 マイページ ログアウト

マイクロデータの利用

● 統計データを探す

開発者向け **API** 他の絞込

LOD

すべて
政府統計一覧の中から探します

分野
17の統計分野から探します

組織
統計を作成した府省等から探します

キーワード検索： 例：国勢調査 検 索

● 統計データを活用する

グラフ
主要指標をグラフで表示
(統計ダッシュボード)

時系列表
主要指標を時系列表で表示
(統計ダッシュボード)

地図
地図上に統計データを表示
(統計GIS)

地域
都道府県、市区町村の
主要データを表示

利用ガイド

● 統計データの高度利用

マイクロデータの利用
公的統計のマイクロデータの利用案内

開発者向け
API、LODで統計データを取得

● 統計関連情報

統計分類・調査項目
統計分類、市区町村コード、調査項目を表示

利用ガイド・マニュアル③



[ホーム](#) [機能概要 ▾](#) [開発支援情報 ▾](#) [イベント ▾](#)

[トップページ](#) / [機能概要](#) / [API仕様](#)

 **機能概要**

- 利用ガイド
- API仕様**
- 提供データ
- 統計分野
- クレジット表示
- 開発支援情報

 **関連リンク**



政府統計の総合窓口



総務省統計局
Statistics Japan



独立行政法人
統計センター
NSTAC

API仕様

API機能の仕様の詳細は、以下を参照してください。

API仕様・・・2019/ 7/ 26提供開始 NEW

統計データをXML、JSON、CSVで提供する仕様です([Word](#) / [PDF](#) / [HTML](#))

現在のバージョン： 3.0
ホスト名： api.e-stat.go.jp
リクエストURL： http://api.e-stat.go.jp/rest/3.0/app/・・・

[バージョン2.1からの変更点について](#)

API仕様・・・2016/ 7/14提供開始

統計データをXML、JSON、CSVで提供する仕様です([Word](#) / [PDF](#) / [HTML](#))

現在のバージョン： 2.1

API機能のホームページから、「機能概要」メニュー下の「API仕様」を選択するとこの画面が表示される。

利用ガイドを参照する他、API仕様を入手しておくといい。

II. 取得データの特定



家計調査

全国約9千世帯を対象として、家計の収入・支出、貯蓄・負債などを毎月調査している。

詳しくは総務省統計局の調査サイトへ。

<https://www.stat.go.jp/data/kakei/index.html>

データ特定①

e-Stat

政府統計の総合窓口

統計で見る日本

e-Statは、日本の統計が閲覧できる政府統計ポータルサイトです

お問い合わせ | ヘルプ | English

統計データを探す

統計データの活用

統計データの高度利用

統計関連情報

リンク集

分野から探す

組織から探す

データベース

ファイル

国土・気象

人口・世帯

労働・賃金

農林水産業

鉱工業

商業・サービス業

企業・家計・経済

住宅・土地・建設

エネルギー・水

運輸・観光

情報通信・科学技術

教育・文化・スポーツ・生活

行財政

司法・安全・環境

社会保障・衛生

国際

その他

国土・気象



> すべて見る

e-Stat [<https://www.e-stat.go.jp/>] から、
「統計データを探す」⇒「分野から探す」⇒
「企業・家計・経済」を選択

労働・賃金



主な調査

- > 労働力調査
- > 就業構造基本調査
- > 民間給与実態統計調査
- > 毎月勤労統計調査
- > 賃金構造基本統計調査
- > 船員労働統計調査

> すべて見る (68 調査)

農林水産業



主な調査

- > 農業経営統計調査
- > 農林業センサス
- > 漁業センサス
- > 作物統計調査
- > 海面漁業生産統計調査
- > 木材統計調査
- > 牛乳乳製品統計調査

> すべて見る (72 調査)

鉱工業

商業・サービス業

企業・家計・経済

住宅・土地・建設

データ特定②



政府統計の総合窓口

統計で見る日本

e-Statは、日本の統計が閲覧できる政府統計ポータルサイトです

[お問い合わせ](#) | [ヘルプ](#) | [English](#)

[統計データを探す](#) | [統計データの活用](#) | [統計データの高度利用](#) | [統計関連情報](#) | [リンク集](#)

[トップページ](#) / [統計データを探す](#) / [すべて](#)

選択条件: [企業・家計・経済](#) > [家計](#) ✕

[政府統計一覧に戻る \(すべて解除\)](#)

8 調査のデータ
(45,645 件のデータセット)

データ種別

 データベース 5

 ファイル 8

統計分野 (小分類) で絞り込み ✕

企業・家計・経済
> 家計 [8]

統計分野 (大分類) で絞り込み ▼

統計分野 (小分類) で絞り込み ▼

組織で絞り込み ▼

統計の種類で絞り込み ▼

政府統計名で絞り込み ▼

提供国期で絞り込み ▼

政府統計 ▼ キーワードを入力 ✕ 🔍 [検索条件の保存](#)

[検索のしかた](#) ⓘ

保存した検索条件

政府統計一覧

「統計分野 (小分類) で絞り込み」
⇒ 「家計」を選択し、表示される
一覧から「家計調査」を選択

政府統計コード ▼	政府統計名 ▼	説明を表示
▶ 00100405	消費動向調査	ⓘ
▶ 00200561	家計調査	ⓘ
▶ 00200563	貯蓄動向調査	
▶ 00200564	全国消費実態調査	ⓘ
▶ 00200565	家計消費状況調査	ⓘ
▶ 00200566	全国単身世帯収支実態調査	ⓘ
▶ 00200567	消費動向指数	ⓘ
▶ 00200568	家計消費単身モニター調査	ⓘ

16

データ特定③

下の画面のデータベースのアイコンをクリックし、次に表示される一覧の「家計収支編」下にある「二人以上の世帯」の「年次」を選択



政府統計の総合窓口

統計で見る日本

e-Statは、日本の統計が閲覧できる政府統計ポータルサイトです

[お問い合わせ](#) | [ヘルプ](#) | [English](#)

[統計データを探す](#) | [統計データの活用](#) | [統計データの高度利用](#) | [統計関連情報](#) | [リンク集](#)

[トップページ](#) / [統計データを探す](#) / [すべて](#)

選択条件: [企業・家計・経済 > 家計](#) × / [家計調査](#) ×

[政府統計一覧に戻る \(すべて解除\)](#)

21,956 件のデータ

データ種別

 データベース 104

 ファイル 21,852

統計分野 (小分類) で絞り込み

[企業・家計・経済 > 家計](#) [21,956] ×

[政府統計名で絞り込み](#) ×

[家計調査 \[21,956\]](#) ×

統計分野 (大分類) で絞り込み

データセット

キーワードを入力

×

Q

検索条件の保存

▼ 検索オプション

☒ 提供分類、表題を検索 ☒ データベース、ファイル内を検索

保存した検索条件

家計調査

一覧形式で表示

家計調査は、統計理論に基づき選定された全国約9千世帯を対象として、家計の収入・支出、貯蓄・負債などを毎月調査しています。

家計調査の結果は、我が国の景気動向の把握、生活保護基準の検討などの基礎資料として利用のほか、地方公共団体、民間の会社などでも利用されています。

二人以上の世帯の結果は、主に、地域・世帯属性ごとに1世帯当たり1か月間の収支金額にまとめ毎月公表、単身世帯及び総世帯の家計収支に関する結果並びに二人以上の世帯の貯蓄・負債に関する結果を四半期ごとに公表しています。

家計調査	データベース	件数 更新日	ファイル	件数 更新日	説明を 表示
家計調査		104件 2020-12-08		21,852件 2020-12-08	

17

データ特定④

「家計収支編」から「二人以上の世帯」の
「年次」を選択するとこの画面が表示される

データセット一覧

[戻る](#)

一覧形式で表示

政府統計名	家計調査
提供統計名	家計調査
提供分類1	家計収支編
提供分類2	二人以上の世帯
提供周期	年次

品目分類の010表を使用するので、この欄にあるAPIのボタンを押し、APIリクエストURLから「statsDataId=」以下に表示される数値をスクレイピングに使用する

表番号	統計表	調査年月	公開（更新）日	表示・ダウンロード
用途分類				
001	用途分類（総数）	-	2020-02-07	→ DB API
002	用途分類（年間収入五分位階級別）	-	2020-02-07	→ DB API
003	用途分類（世帯主の定期収入五分位階級別）			
004	用途分類（世帯人員別）			
005	用途分類（世帯主の年齢階級別）			
006	用途分類（世帯主の職業別）			
008	用途分類（世帯主の勤め先企業規模別）	-	2020-02-07	→ DB API
009	用途分類（住居の所有関係別）	-	2020-02-07	→ DB API
品目分類				
010	品目分類（平成27年改定）（総数：金額）	-	2020-02-07	→ DB API
011	品目分類（平成27年改定）（総数：数量）	-	2020-02-07	→ DB API
012	品目分類（平成27年改定）（年間収入五分位階級：金額）	-	2020-02-07	→ DB API
013	品目分類（平成27年改定）（年間収入五分位階級：数量）	-	2020-02-07	→ DB API

API リクエスト URL

クエリー

http://api.e-stat.go.jp/rest/2.1/app/getStatsData?appId=&lang=J&statsDataId=0003125169&metaGetFlg=Y&cntGetFlg=N§ionHeaderFlg=1

閉じる

III. Rによるスクレイピング

使用パッケージ: RCurl, jsonlite



コード①: スクレイピング

```
setwd("C:/test")    # ファイルを収録するディレクトリを指定する
#RCurlで読み込み、jsonliteパッケージを使ってjson形式のデータを処理する
require(RCurl)
require(jsonlite)
yourAPPID <- "*****" # 各自取得したe-StatのアプリケーションID
# 変数urlにAPIのコマンドを作成する
url <- paste("http://api.e-stat.go.jp/rest/3.0/app/json/getStatsData?appId=",
             yourAPPID,                               # 上でセットした各人のアプリケーションID
             "&lang=J",
             "&statsDataId=0003125169",              # 取得する表のID(半角)
             "&metaGetFlg=Y",
             "&cntGetFlg=N",
             "&sectionHeaderFlg=1", sep="")
dt0 <- getURL(url)    # ちょっと時間がかかる
dt1 <- fromJSON(dt0)
str(dt1, max.level=3)                                # 出力1
str(dt1$GET_STATS_DATA$STATISTICAL_DATA$RESULT_INF) # 出力2
str(dt1$GET_STATS_DATA$STATISTICAL_DATA$DATA_INF$VALUE) # 出力3
str(dt1$GET_STATS_DATA$STATISTICAL_DATA$CLASS_INF$CLASS_OBJ$CLASS)
# 出力4
```

取得データの全体構造（出力1）

```
> str(dt1, max.level=3)
List of 1
 $ GET_STATS_DATA:List of 3  ..$ RESULT      :List of 3
   ...$ STATUS   : int 0
   ...$ ERROR_MSG: chr "正常に終了しました。"
   ...$ DATE     : chr "2020-11-04T23:55:49.966+09:00"
   ..$ PARAMETER :List of 7
   ...$ LANG     : chr "J"
   ...$ STATS_DATA_ID : chr "0003125169"
   ...$ DATA_FORMAT : chr "J"
   ...$ START_POSITION : int 1
   ...$ METAGET_FLG   : chr "Y"
   ...$ CNT_GET_FLG   : chr "N"
   ..$ SECTION_HEADER_FLG: int 1
   ..$ STATISTICAL_DATA:List of 4
     ...$ RESULT_INF:List of 4
     ...$ TABLE_INF:List of 17
     ...$ CLASS_INF :List of 1
     ...$ DATA_INF  :List of 2
```

エラーはここに表示される

← この中も確認する必要がある

← この下にメタ情報がある

← この下に統計数値が収録されている



統計数値: dt1\$GET_STATS_DATA\$STATISTICAL_DATA\$DATA_INF\$VALUE

メタ情報: dt1\$GET_STATS_DATA\$STATISTICAL_DATA\$CLASS_INF\$CLASS_OBJ\$CLASS

データの取得状況（出力2）

```
> str(dt1$GET_STATS_DATA$STATISTICAL_DATA$RESULT_INF)
```

List of 4

```
$ TOTAL_NUMBER: int 535409
```

```
$ FROM_NUMBER : int 1
```

```
$ TO_NUMBER   : int 100000
```

```
$ NEXT_KEY    : int 100001
```

ここから、要求した表の数値情報が535409個存在し、先ほど行ったデータ取得により、このうちの冒頭の10万個を取得したことを示している。このため、全体を取得するためには、あと5回データ取得を繰り返す必要があることがわかる。

統計数値（出力3）

データフレームの1レコードに統計数値が1つとその属性コードが6種類含まれる

```
> str(dt1$GET_STATS_DATA$STATISTICAL_DATA$DATA_INF$VALUE) # 出力3
```

```
'data.frame': 100000 obs. of 7 variables:      ← 10万データ取得
```

```
$ @tab : chr "01" "01" "01" "01" ...
```

```
$ @cat01: chr "000100000" "000100000" "000100000" "000100000" ...
```

```
$ @cat02: chr "03" "03" "03" "03" ...
```

```
$ @area : chr "00000" "00000" "00000" "00000" ...
```

```
$ @time : chr "20000000000" "20010000000" "20020000000" "20030000000" ...
```

```
$ @unit : chr "一万分比" "一万分比" "一万分比" "一万分比" ...
```

```
$ $ : chr "10000" "10000" "10000" "10000" ...      ← 統計数値
```

属性コード

統計数値の属性情報（出力4）

```
> str(dtl$GET_STATS_DATA$STATISTICAL_DATA$CLASS_INF$CLASS_OBJ$CLASS)
# 出力4
List of 5
 $ :List of 3
  ..$ @code : chr "01"
  ..$ @name : chr "金額"
  ..$ @level: chr ""
  CLASS[[1]]: @tab

 $ :'data.frame':    703 obs. of  5 variables:
  ..$ @code   : chr [1:703] "000100000" "000200000" "000300000" "000400000" ...
  ..$ @name   : chr [1:703] "世帯数分布(抽出率調整)" "集計世帯数" "世帯人員" "18歳未満人員" ...
  ..$ @level  : chr [1:703] "1" "1" "1" "2" ...
  ..$ @unit   : chr [1:703] "一万分比" "世帯" "人" "人" ...
  ..$ @parentCode: chr [1:703] NA NA NA "000300000" ...
  CLASS[[2]]: @cat01

 $ :'data.frame':    4 obs. of  3 variables:
  ..$ @code : chr [1:4] "03" "04" "01" "02"
  ..$ @name : chr [1:4] "二人以上の世帯(2000年～)" "二人以上の世帯のうち勤労者世帯(2000年～)" "二人以上の世帯(農林漁家世帯を除く)(1985年～2007年,2017年)"
  ..$ @level: chr [1:4] "1" "1" "1" "1"
  CLASS[[3]]: @cat02

 $ :'data.frame':    53 obs. of  3 variables:
  ..$ @code : chr [1:53] "00000" "01003" "02003" "03003" ...
  ..$ @name : chr [1:53] "全国" "01100 札幌市" "02201 青森市" "03201 盛岡市" ...
  ..$ @level: chr [1:53] "1" "1" "1" "1" ...
  CLASS[[4]]: @area

 $ :'data.frame':    35 obs. of  3 variables:
  ..$ @code : chr [1:35] "1985000000" "1986000000" "1987000000" "1988000000" ...
  ..$ @name : chr [1:35] "1985年" "1986年" "1987年" "1988年" ...
  ..$ @level: chr [1:35] "1" "1" "1" "1" ...
  CLASS[[5]]: @time
```

出力3にある属性情報に対応

コード②: 010表データを全て取得

```
# 全体のデータ数をTotal、続きの取得の開始ポイントをNKeyにセットして取得を繰り返す
(Total <- dt1$GET_STATS_DATA$STATISTICAL_DATA$RESULT_INF$TOTAL_NUMBER)
(NKey <- dt1$GET_STATS_DATA$STATISTICAL_DATA$RESULT_INF$NEXT_KEY)
rp1 <- ceiling(Total / (NKey-1))    # 必要な読込回数を計算
nb1 <- 1:rp1
nb2 <- formatC(nb1, width=2, flag="0")    # 2桁でゼロ捕捉
(dname <- apply(cbind("dd", nb2), 1, paste, collapse=""))
# [1] "dd01" "dd02"      ...      # データを収録する変数名をdnameに生成
# 一回目に読み込んだ統計数値部分を最初の領域に保存
assign(dname[1], dt1$GET_STATS_DATA$STATISTICAL_DATA$DATA_INF$VALUE)
for (i in 2:rp1){    # この処理には数分かかる
  url <- paste("http://api.e-stat.go.jp/rest/3.0/app/json/getStatsData?appld=",
    yourAPPID,          # アプリケーションID
    "&lang=J",
    "&statsDataId=0003125169", # 取得する表のID(半角)
    "&startPosition=", NKey,    # データ取得の開始点
    "&metaGetFlg=Y",
    "&cntGetFlg=N",
    "&sectionHeaderFlg=1", sep="")
  dt0 <- getURL(url)    # 同じ領域を再利用
  dt1 <- fromJSON(dt0)
  # 利用する部分だけ確保した領域に保存する
  assign(dname[i], dt1$GET_STATS_DATA$STATISTICAL_DATA$DATA_INF$VALUE)
  # データ取得の開始ポイントを更新
  NKey <- dt1$GET_STATS_DATA$STATISTICAL_DATA$RESULT_INF$NEXT_KEY
}
```


データの抽出条件を決める

- 食料関係の品目
⇒ @cat01が「010000000」(食料)の内訳の最下位品目
- 二人以上の世帯
⇒ @cat02が「03」
- 都道府県庁所在市の47市に着目する
⇒ @areaの2番目から48番目
- 品目改正がないのは直近5年分(2015~2019年)
⇒ @tmが「2015000000~2019000000」(コード末尾5つ)

取得データの内容

コード情報	対応データ	内容
CLASS[1]	@tab	金額のみの一種類
CLASS[2]	@cat01	品目分類等703項目
CLASS[3]	@cat02	世帯類型4種類
CLASS[4]	@area	地域(全国及び52都市)
CLASS[5]	@tm	年次(1985~2019年の35年分)

コード③: 取得データの確認と絞り込み

```
# 前のコードの続きから
# 出力3参照。取得データの全ての属性を確認する。
dt1$GET_STATS_DATA$STATISTICAL_DATA$CLASS_INF$CLASS_OBJ$CLASS[1]
dt1$GET_STATS_DATA$STATISTICAL_DATA$CLASS_INF$CLASS_OBJ$CLASS[2] #品目コード
dt1$GET_STATS_DATA$STATISTICAL_DATA$CLASS_INF$CLASS_OBJ$CLASS[3] # 世帯類型
dt1$GET_STATS_DATA$STATISTICAL_DATA$CLASS_INF$CLASS_OBJ$CLASS[4] # 地域コード
dt1$GET_STATS_DATA$STATISTICAL_DATA$CLASS_INF$CLASS_OBJ$CLASS[5] # 年次コード
# 変数名が長すぎるので使うものだけ名前を付け直す
itm <- as.data.frame(dt1$GET_STATS_DATA$STATISTICAL_DATA$CLASS_INF$CLASS_OBJ$CLASS[2])
cty <- as.data.frame( dt1$GET_STATS_DATA$STATISTICAL_DATA$CLASS_INF$CLASS_OBJ$CLASS[4])
per <- as.data.frame( dt1$GET_STATS_DATA$STATISTICAL_DATA$CLASS_INF$CLASS_OBJ$CLASS[5])
# 品目の選択: 内訳を持たない末端の品目(必ずしもレベル5ではない)を特定する
f.lv <- rep(0, length(itm$X.level)) # 末端フラグ
for (i in 1:(length(itm$X.level)-1)) if (itm$X.level[i] >= itm$X.level[i+1]) f.lv[i] <- 1
# 次の品目と同じレベルなら末端品目と判定する
length(which(f.lv==1)) # [1] 564
cbind(itm$X.name, itm$X.leve, f.lv) # 確認表を表示
# 抽出条件セット -----
s.setai <- "03" # 二人以上の世帯(2000年～)
s.area <- cty[2:48,] # 都道府県庁所在市のみ
s.itm <- itm[which(f.lv==1 & itm$X.code > "010000000" & itm$X.code < "020000000"),]
s.tm <- per[31:35,] # コード表末尾の5年分
```

コード④: データ選択と整形

```
# 表作成 -----
tb0 <- array(NA, c(length(s.area$X.code), length(s.itm$X.code), length(s.tm$X.code)))

# 市名の頭にある地域コードを除去
dimnames(tb0)[[1]] <- substr(s.area$X.name, 6, length(s.area$X.name))
# 品目名の頭にある品目番号を除去
require(stringi) # 文字列操作
bk.pos <- stri_locate_first_regex(s.itm$X.name, " ") # 探索文字列の開始位置と終了位置が戻る
dimnames(tb0)[[2]] <- substr(s.itm$X.name, bk.pos[1], length(s.itm$X.name))
dimnames(tb0)[[3]] <- s.tm$X.name
dim(tb0)           # [1] 47都道府県庁所在市 213 品目 5 時点(年)
```

- 作成する表イメージの枠をあらかじめ三次元配列tb0として作成し、地域名、品目名、年次をセットしている
- 地域名には5文字の地域コードと空白が付与されているので6文字目から使用している
- 品目名には文字数の変わる品目コードと空白が付与されているため、最初に出現する空白を検出し、その空白以降の文字を使用している

コード④: データ選択と整形 (続き)

```
# 抽出条件に合う統計数値のみtb0に収録する
f.err <- matrix(NA, nr=rp1, nc=3) # 確認用エラーフラグ
for (i in 1:rp1){
  dat0 <- get(dname[i])
  colnames(dat0) <- c("tab", "cat01", "cat02", "area", "time", "unit", "val")
  # 元の変数名は@がつくが、Rの変数名は記号から始まってはいけない
  dat1 <- subset(dat0, (cat01 %in% s.itm$X.code) & (cat02 == s.setai)
                 & (area %in% s.area$X.code) & (time %in% s.tm$X.code))

  c1 <- match(dat1$area, s.area$X.code)
  c2 <- match(dat1$cat01, s.itm$X.code)
  c3 <- match(dat1$time, s.tm$X.code)
  f.err[i,1] <- length(which(is.na(c1)))
  f.err[i,2] <- length(which(is.na(c2)))
  f.err[i,3] <- length(which(is.na(c3)))

  # tb0[c1,c2,c3] <- as.numeric(dat1$val) としたいが時間がかかる
  for (j in 1:dim(dat1)[1]){
    tb0[c1[j],c2[j],c3[j]] <- as.numeric(dat1$val[j])
  }
}
f.err      # 全て0であることを確認
apply(apply(tb0, c(1,2), is.na), 1, sum) # 年次別の欠測数チェック
# 2015年 2016年 2017年 2018年 2019年
#   0     0     0     0     0
```

参考

スクレイピングの時点で取得するデータを絞ることもできる [仕様書参照]

```
url <- paste("http://api.e-  
stat.go.jp/rest/3.0/app/json/getStatsData?appId=",  
  yourAPPID, # 上でセットした各人のアプリケーションID  
  "&lang=J",  
  "&statsDataId=0003|25|69", # 取得する表のID(半角)  
  "&cdTimeFrom=2015000000", # 時系列を2015年から  
  "&cdAreaFrom=3",  
  "&metaGetFlg=N",  
  "&cntGetFlg=N",  
  "&sectionHeaderFlg=1", sep="")  
dt0 <- getURL(url)  
dt1 <- fromJSON(dt0)
```

IV. 分析1： 都道府県庁所在市の食料消費

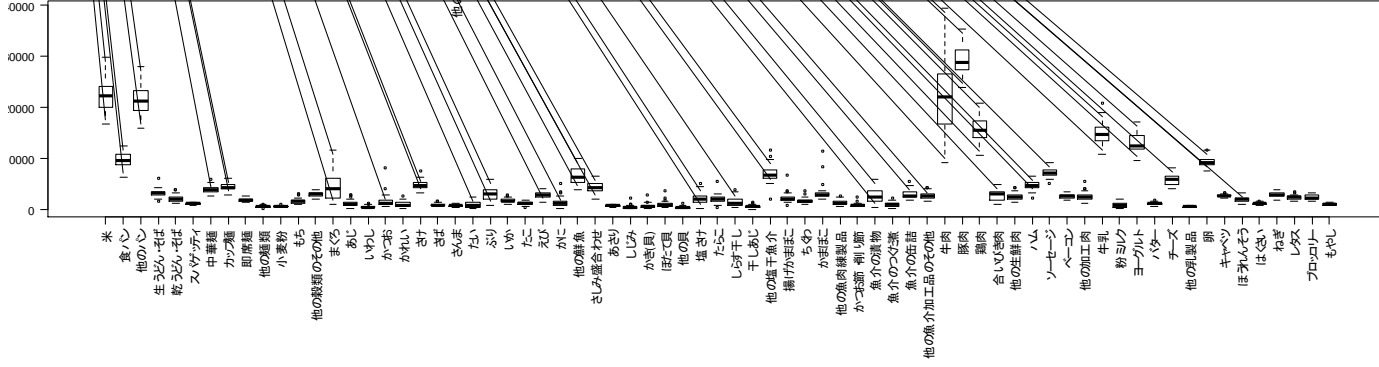
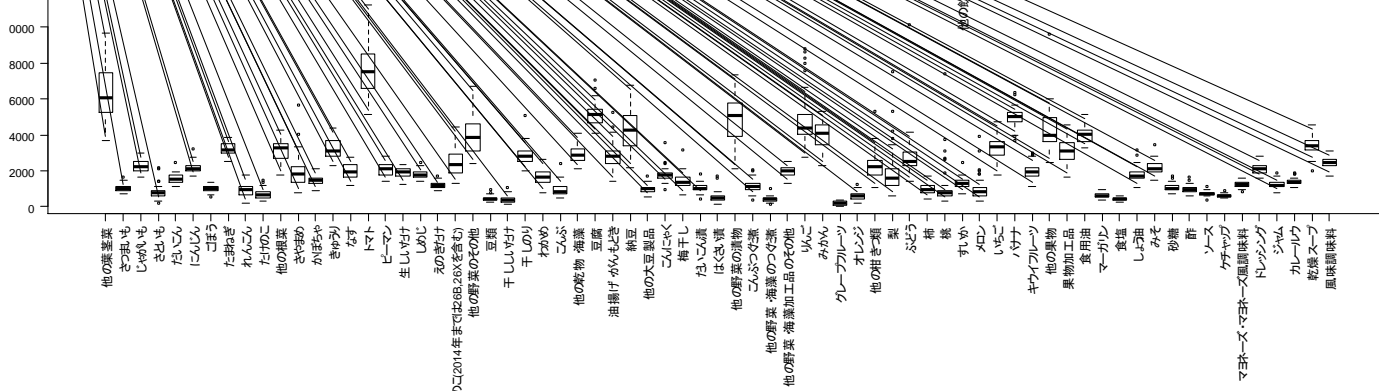
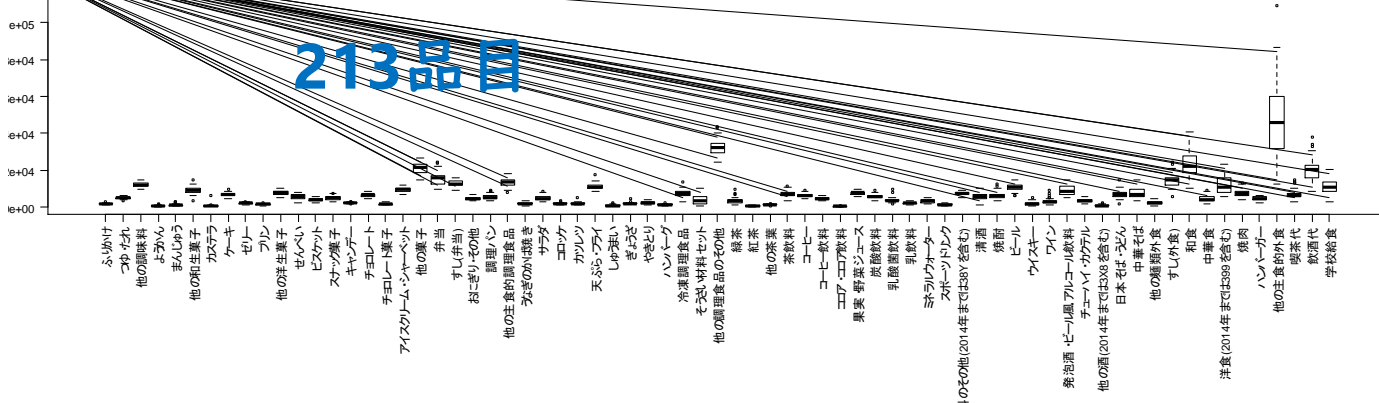


作業手順

1. 35年分の時系列データから、都道府県庁所在市・消費支出品目について欠測のない直近5年分(2015~2019年)のデータを使用し、特に飲食関係の品目に着目する
2. 品目は階層構造を持っているので、最下位レベルの品目のみを抽出する
3. 都道府県庁所在市・消費支出品目別に平均値(A)と中央値(B)をとる
4. 品目により価格差があるため標準化(ただしAは平均値と標準偏差、Bは中央値と中央絶対偏差による)
5. 都市間のユークリッド距離を算出し、Ward法でクラスター分析をする

品目一覧 (2019年データ)

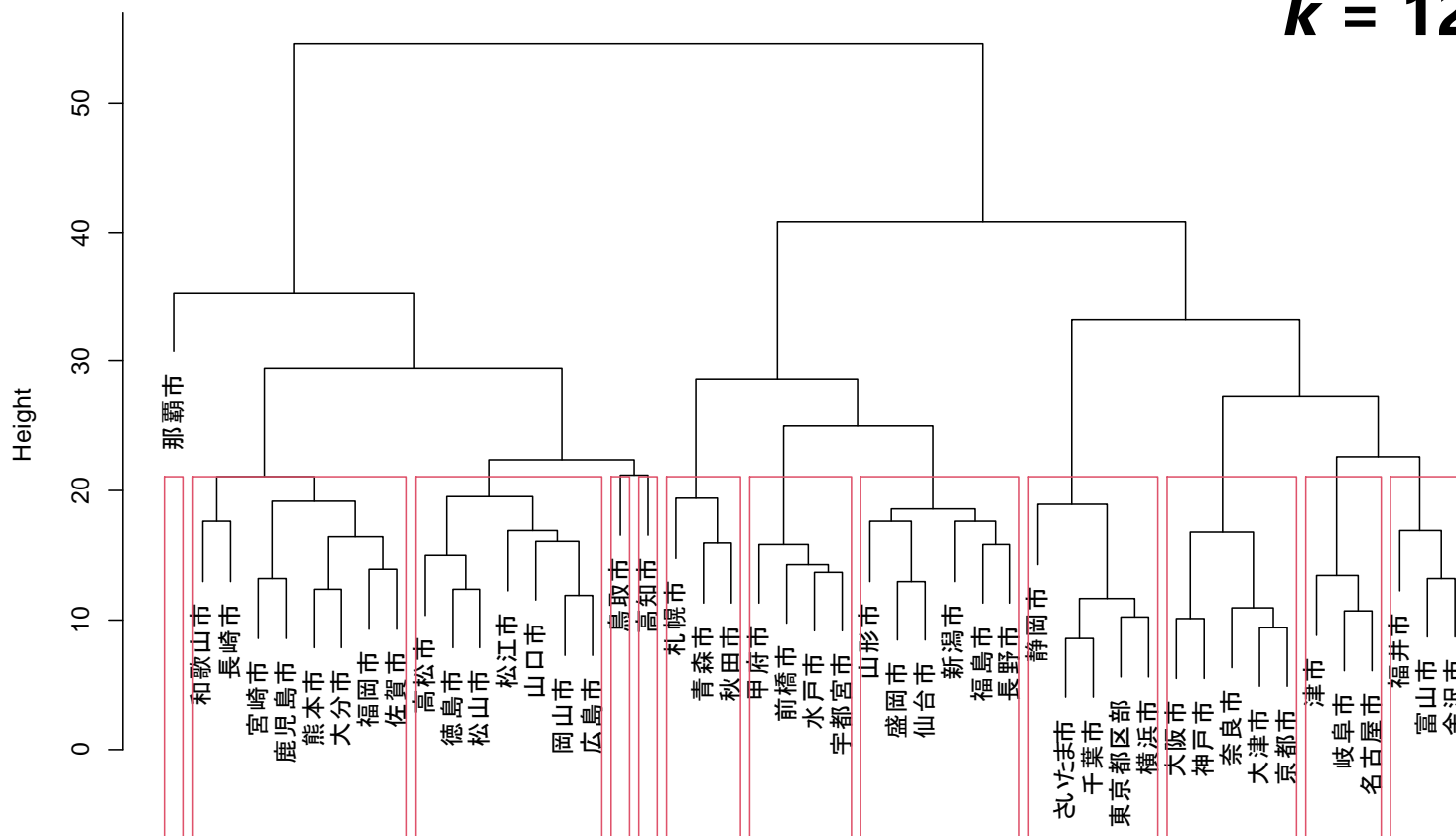
213品目



五年分の平均値による分類

Cluster Dendrogram

$k = 12$

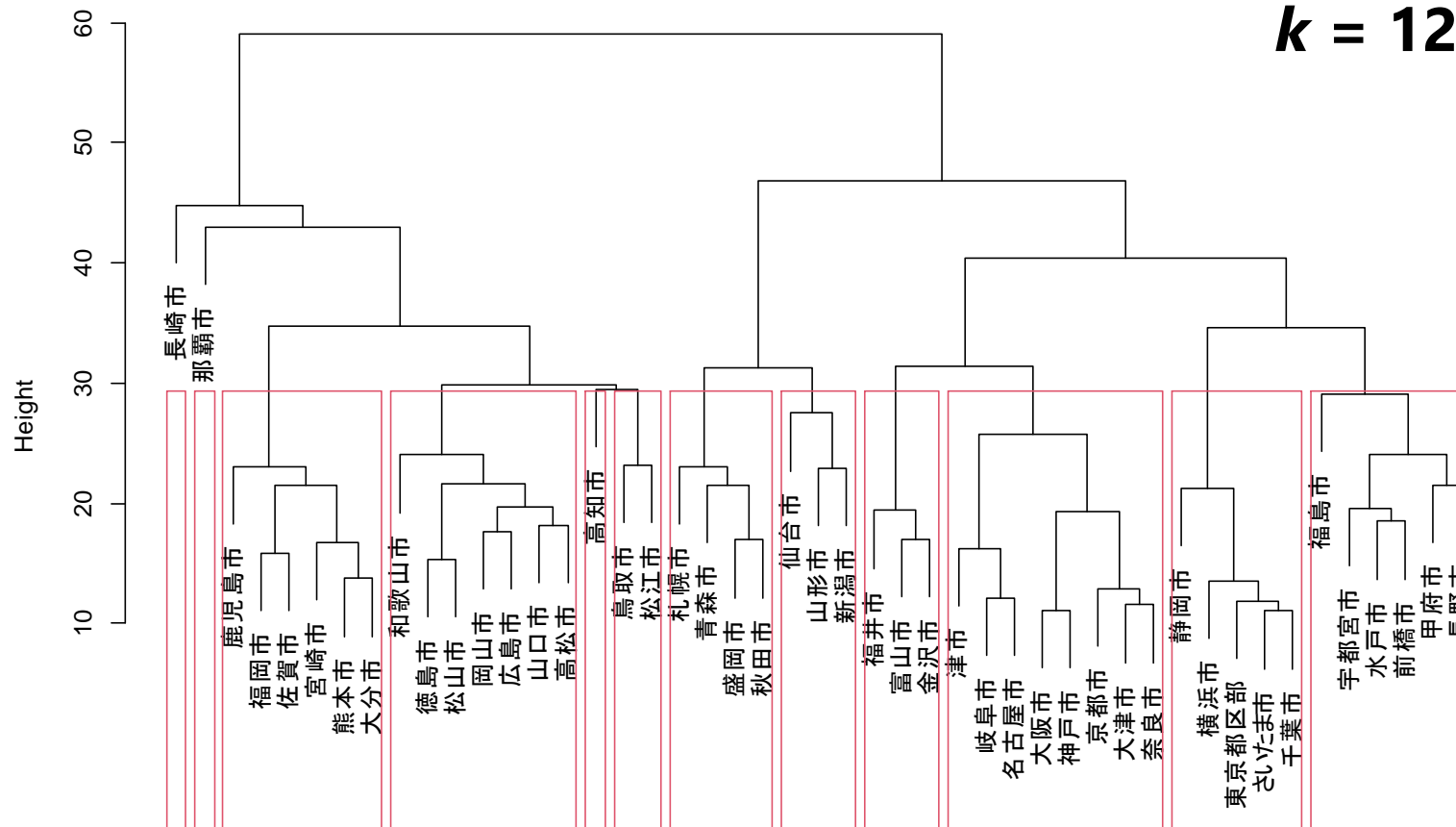


Mn5.d
hclust (*, "ward.D2")

五年分の中央値による分類

Cluster Dendrogram

$k = 12$



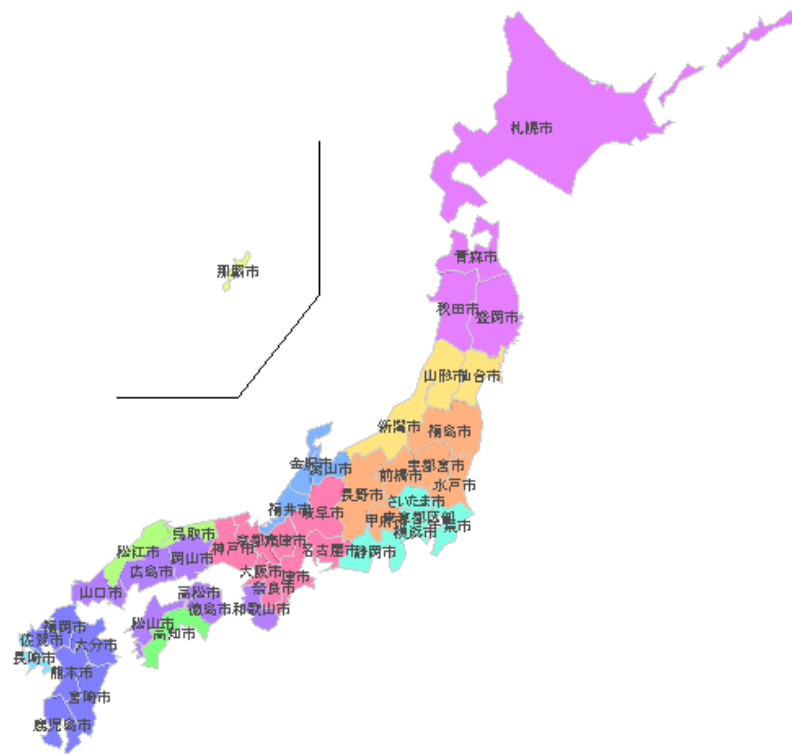
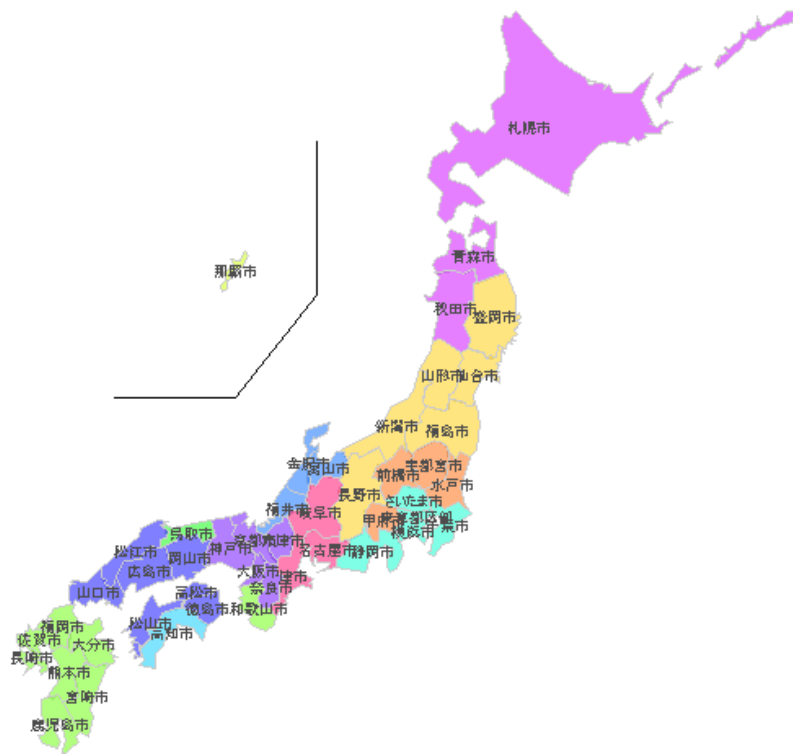
Md5.d
hclust(*, "ward.D2")

分類に基づく色分け ($k=12$)

分類結果は地理的な位置の影響が大きい

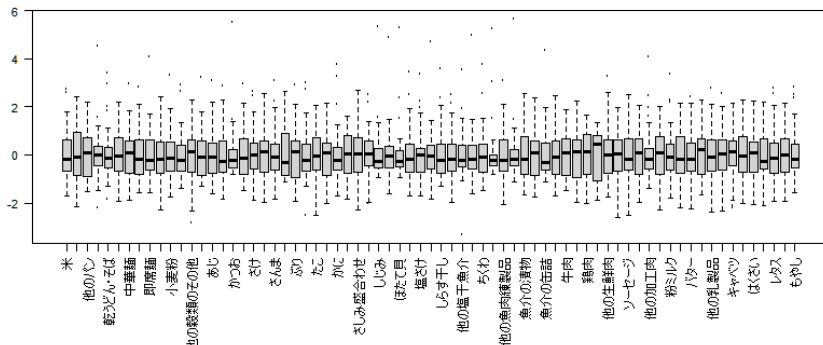
平均値

中央値

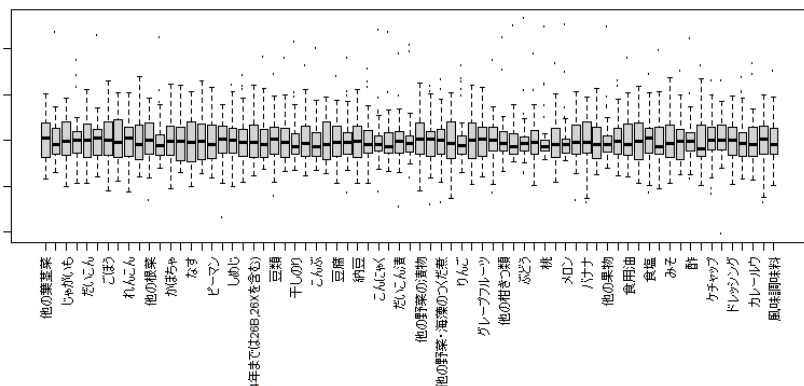


外れ値チェック

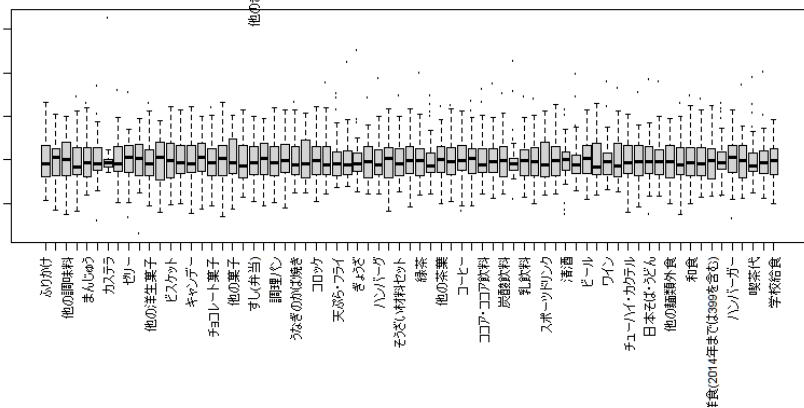
標準化平均值(1)



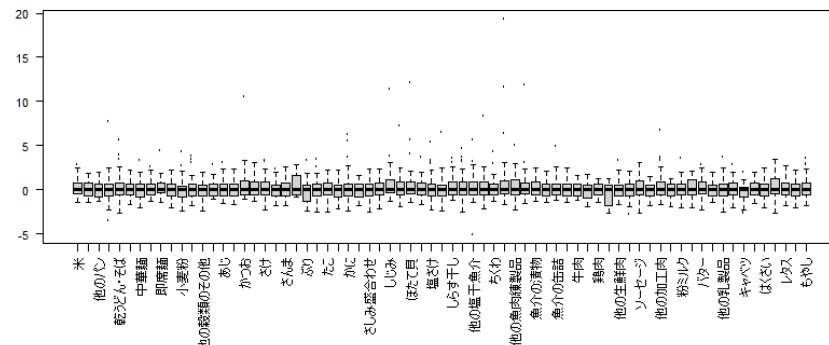
標準化平均値(2)



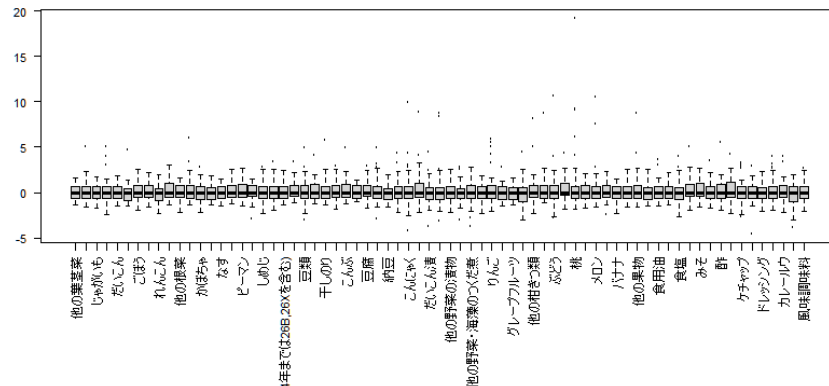
標準化平均值(3)



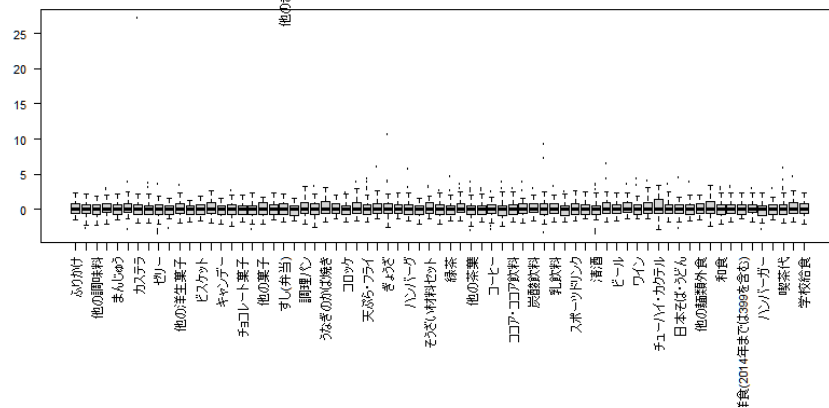
標準化中央値(1)



標準化中央値(2)

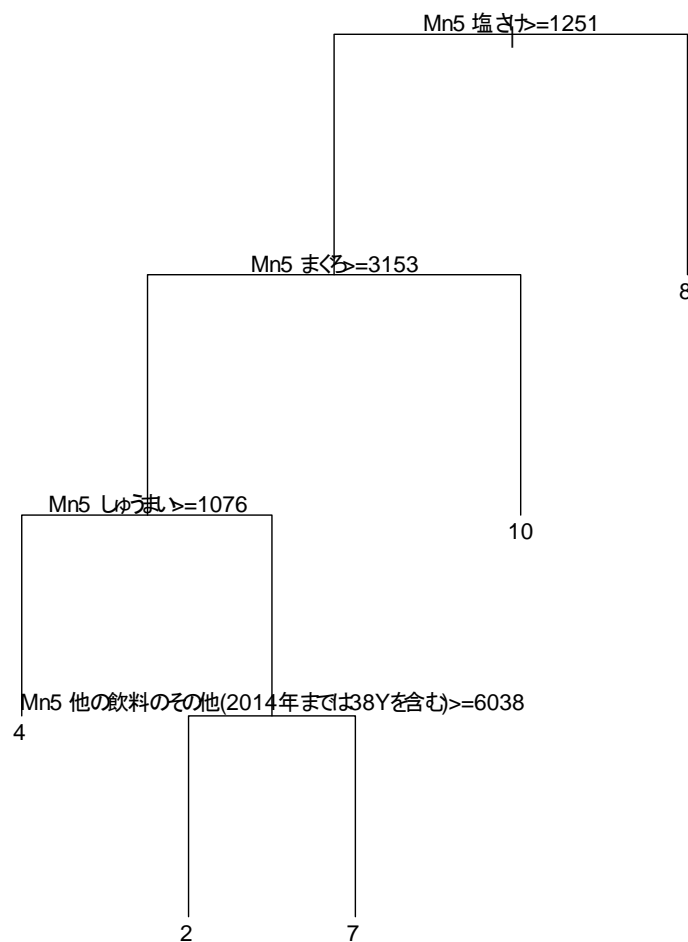


標準化中央値(3)

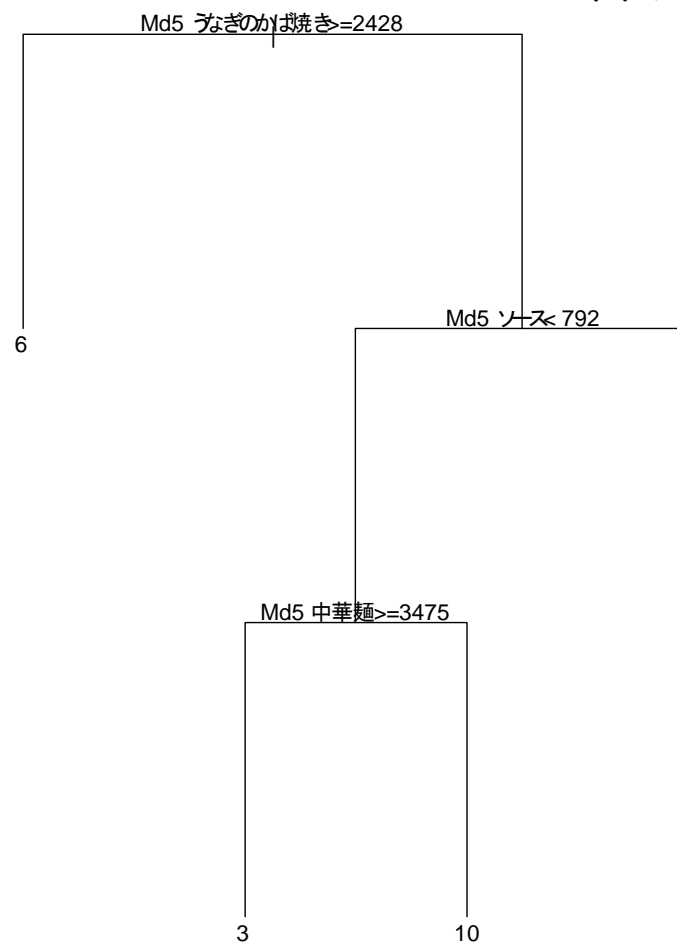


分類に影響の大きい品目

(平均値の場合)

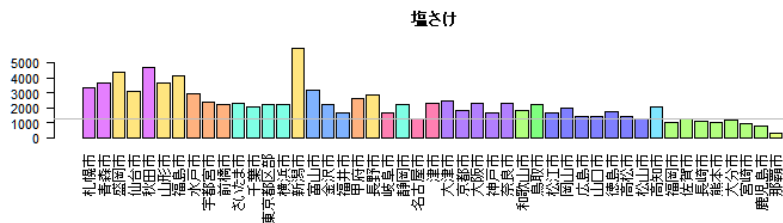


(中央値の場合)

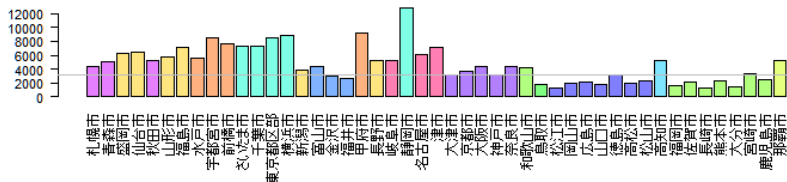


影響の大きい品目の状況

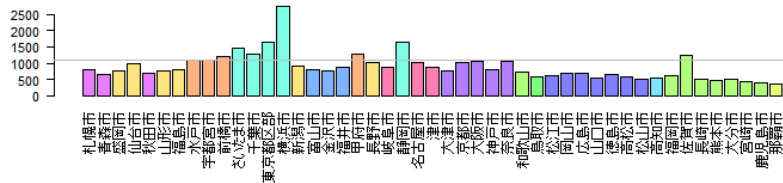
平均値データ



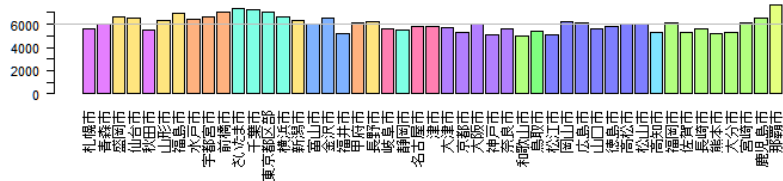
まぐろ



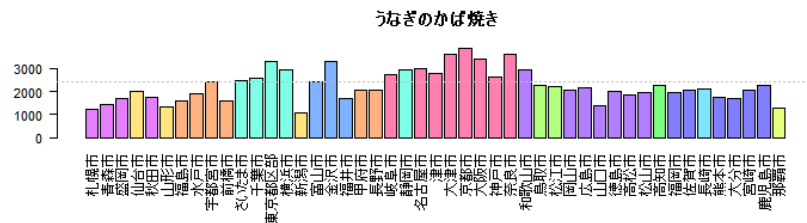
しゅうまい



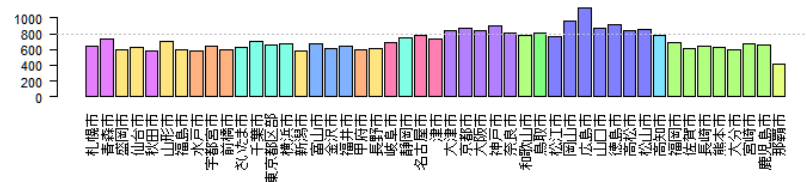
他の飲料のその他(2014年までは38Yを含む)



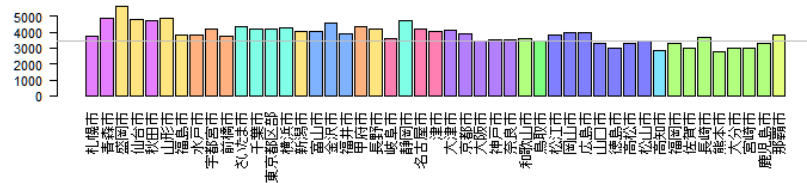
中央値データ



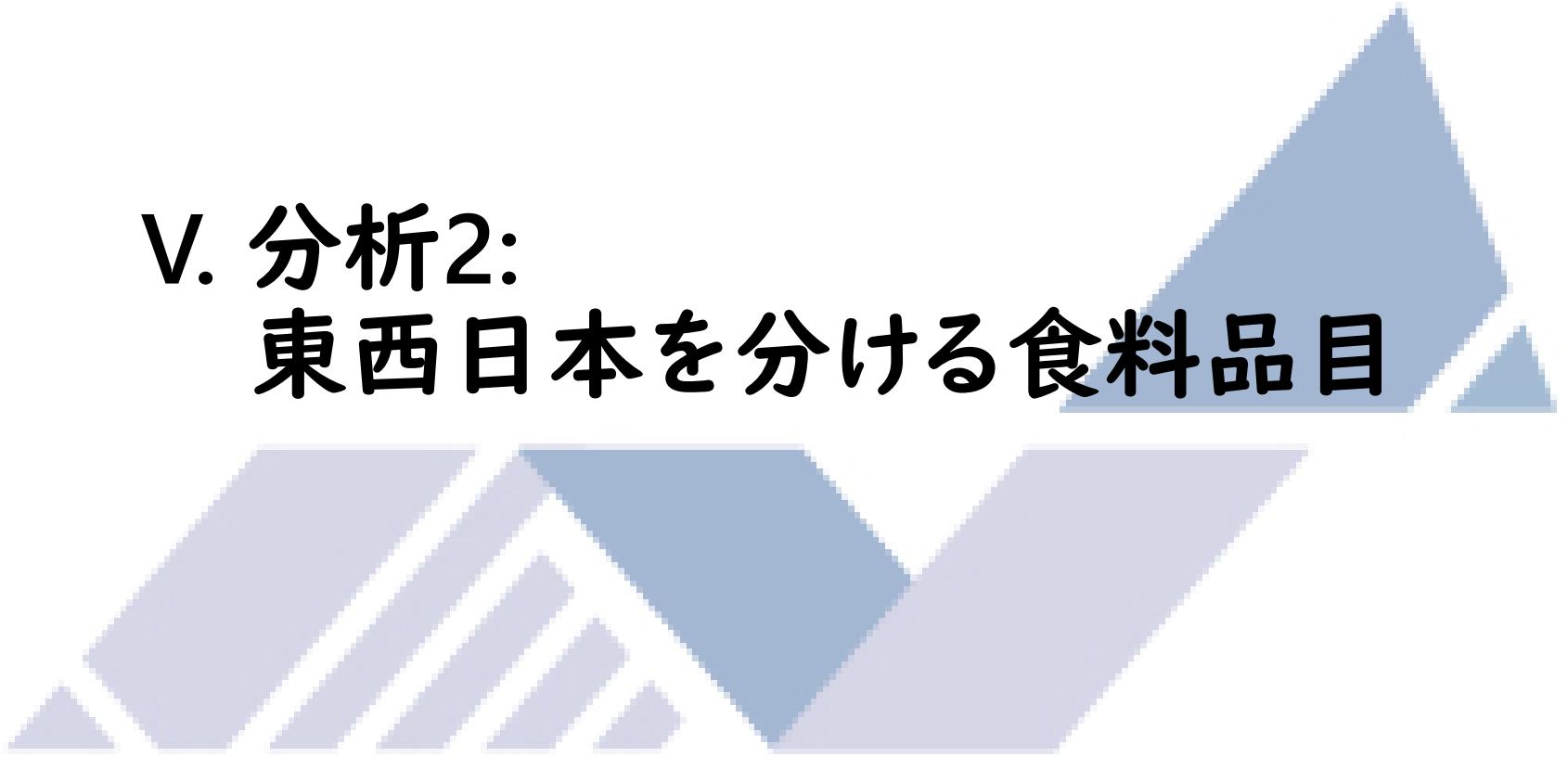
ソース



中華麵



V. 分析2: 東西日本を分ける食料品目



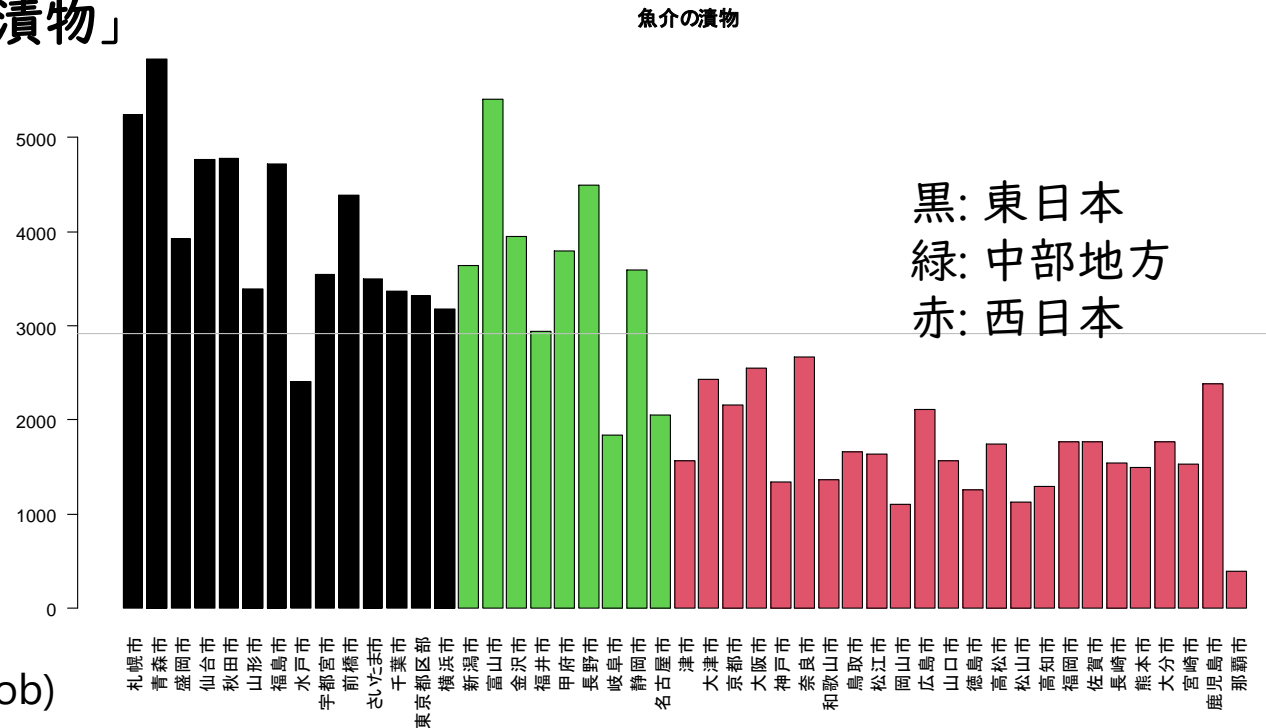
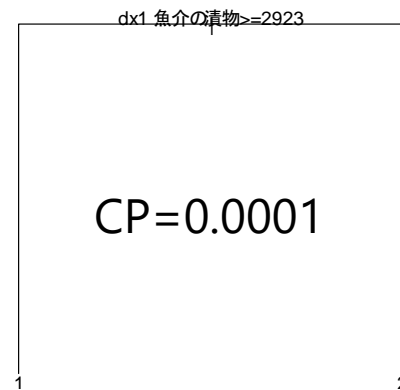
東日本と西日本(2019)

東西を分ける品目を分類木(CART)で探してみた

目的変数: 東日本(1)・西日本(2)の別 (中部地方は除外)

説明変数: 食料の全最下位品目

結果: 「魚介の漬物」



n = 38

node), split, n, loss, yval, (yprob)

* denotes terminal node

1) root 38 14 2 (0.3684211 0.6315789)

2) dx1 魚介の漬物 >= 2923 13 0 1 (1.0000000 0.0000000) *

3) dx1 魚介の漬物 < 2923 25 1 2 (0.0400000 0.9600000) *

東日本と西日本（2015～2019）

2015年 n= 38 node), split, n, loss, yval, (yprob)

1) root 38 14 2 (0.36842105 0.63157895)

2) dx1[, i] 納豆 >= 3886 15 1 1 (0.93333333 0.06666667) *

3) dx1[, i] 納豆 < 3886 23 0 2 (0.00000000 1.00000000) *

2016年 n= 38 node), split, n, loss, yval, (yprob)

1) root 38 14 2 (0.3684211 0.6315789)

2) dx1[, i] 魚介の漬物 >= 3339.5 13 0 1 (1.0000000 0.0000000) *

3) dx1[, i] 魚介の漬物 < 3339.5 25 1 2 (0.0400000 0.9600000) *

2017年 n= 38 node), split, n, loss, yval, (yprob)

1) root 38 14 2 (0.36842105 0.63157895)

2) dx1[, i] 合いびき肉 < 2899 15 1 1 (0.93333333 0.06666667) *

3) dx1[, i] 合いびき肉 >= 2899 23 0 2 (0.00000000 1.00000000) *

2018年 n= 38 node), split, n, loss, yval, (yprob)

1) root 38 14 2 (0.36842105 0.63157895)

2) dx1[, i] 魚介の漬物 >= 3052.5 15 1 1 (0.93333333 0.06666667) *

3) dx1[, i] 魚介の漬物 < 3052.5 23 0 2 (0.00000000 1.00000000) *

2019年 n= 38 node), split, n, loss, yval, (yprob)

1) root 38 14 2 (0.3684211 0.6315789)

2) dx1[, i] 魚介の漬物 >= 2923 13 0 1 (1.0000000 0.0000000) *

3) dx1[, i] 魚介の漬物 < 2923 25 1 2 (0.0400000 0.9600000) *

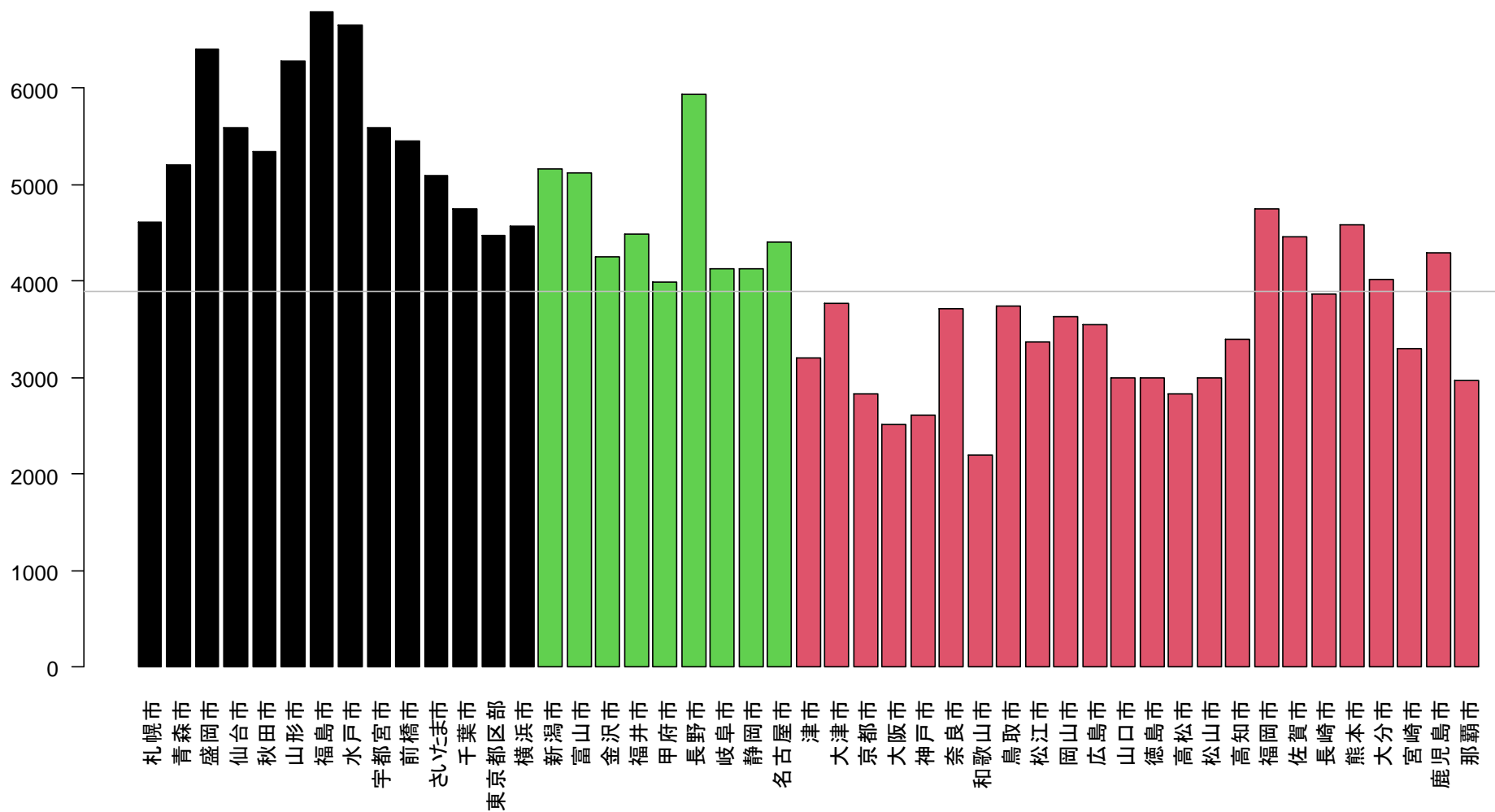
2015年	納豆
2016年	魚介の漬物
2017年	合いびき肉
2018年	魚介の漬物
2019年	魚介の漬物

「魚介の漬物」の定義

魚介類のみそ漬、しょう油漬、味りん漬、あわ漬、酢漬、糠漬、かす漬及びマリネ。うの花漬、しめさば、松前漬、うみたけ(貝)のかす漬、ぬかいわし、ぬかにしん、酢だこ、味付たこ

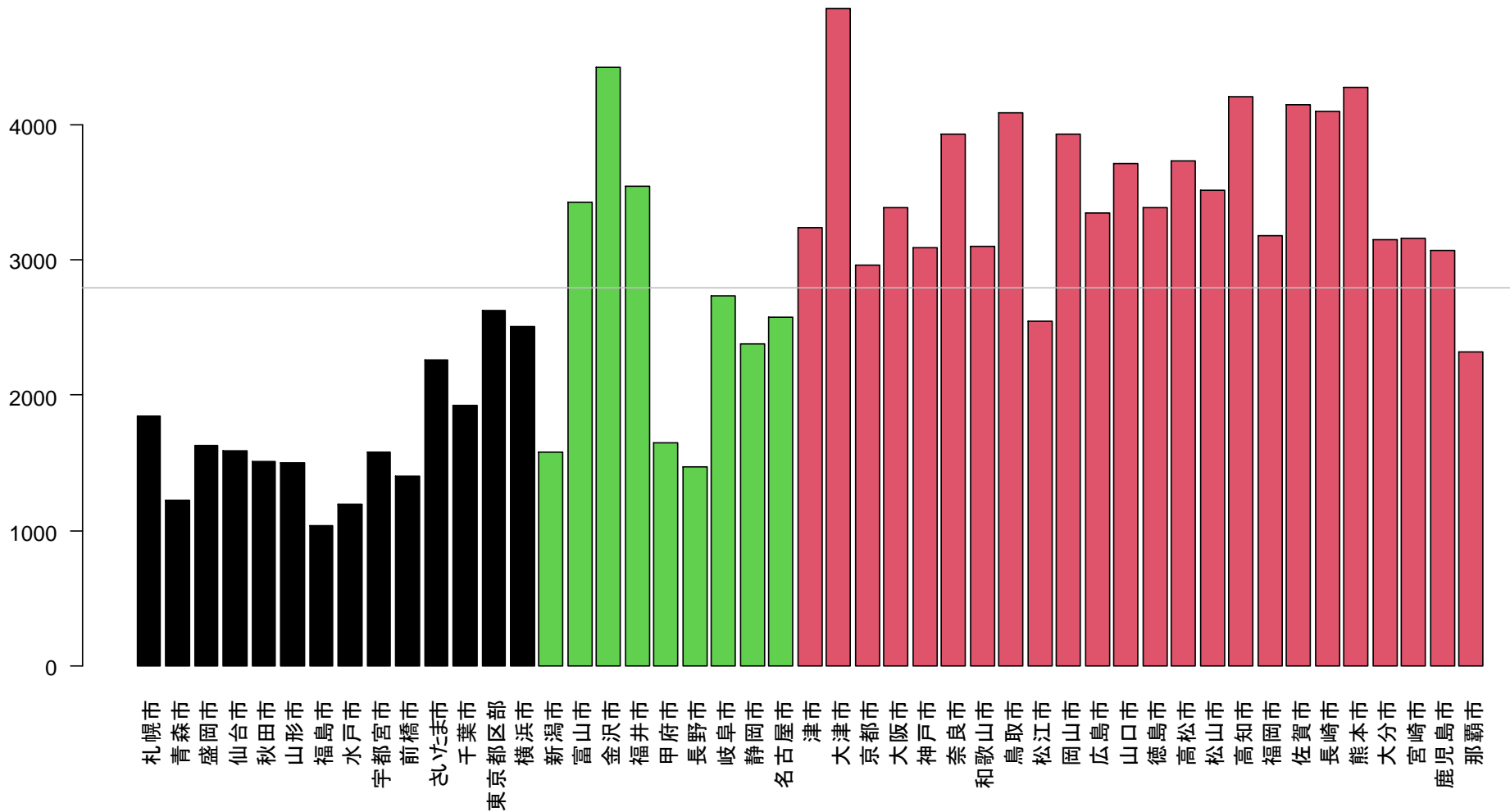
納豆(2019)

納豆



合いびき肉(2019)

合いびき肉



まとめ

- e-STAT APIは慣れると便利
- csvによるデータ取得よりもメタ情報を取得できる
Json形式がお勧め
- 家計調査データは面白い
- 政府統計調査にご協力を!

