Applied Scaling & Classification Techniques in Political Science

Analyzing political texts manually: Party programmes and the CMP project
References


Party programmes

If we want to estimate the policy positions of politicians, one of the main source of information at our disposal is political text.

There are of course many different types of political text as we have already seen, but one authoritative source of information about the stated electoral policy positions of political parties is the official party manifesto.
Party programmes

It might be argued that very few real voters read any party manifesto at all, while almost no sane (!!!) voter checks all party manifestoes and conducts an in-depth comparative analysis of these, basing her voting decision on the results of this analysis.

Nonetheless the party manifesto is the official statement of party policy, to which the party can be held accountable – by critics, journalists and expert observers of the political scene.
Party programmes

In this sense, positions outlined in the party manifesto can be taken as “official” party policy.

Moreover, we can take party manifesto as an indicator of the party’s policy preferences at a given point in time (i.e., a perfect indicator to estimate something that happens AFTER that moment, i.e., the kind of cabinet that is going to be formed, its policy program, etc.)
CMP’s objective

How to analyze party manifestos? Well, we could use some of the algorithms discussed up to now (and that we will discuss also later on), or we can rely on some projects that have already done it for us…

The longstanding **Comparative Manifestos Project** (CMP) ([https://manifestoproject.wzb.eu/](https://manifestoproject.wzb.eu/)) has conducted a systematic analysis of party manifestos over a long period of time, using trained **human readers** to code, into a **predefined 56-category coding scheme** (57 if we include the “uncoded” category), **every sentence** of every manifesto investigated
CMP’s objective

Coverage extends to almost every party manifesto issued at every democratic election since World War 2

This has generated a time series of the electoral party policy positions that spans the post-war era for most parties in most democratic states (OECD members, Central and Eastern Europe, Latin America and South-East Asia)
Two-step process

1. **Unitising** – cutting text in **quasi-sentence**

The coding unit is a quasi-sentence, that contains exactly one statement

e.g.: “We need to address our close ties with our neighbours (*1st quasi-sentence*) / as well as the unique challenges facing small business owner in this of economic hardship (*2nd quasi-sentence*)”
Two-step process

2. **Coding** – find the right code for a quasi-sentence: attribute to each coding-unit *one, and only one*, category

CMP developed a category system composed of 56 categories, grouped in 7 policy areas, designed to be comparable between parties, countries, elections and across time.
Table 1: 56 Standard Policy Preferences in Seven Policy Domains

**Domain 1: External Relations**
- 101 Foreign Special Relationships: Positive
- 102 Foreign Special Relationships: Negative
- 103 Anti-Imperialism: Positive
- 104 Military: Positive
- 105 Military: Negative
- 106 Peace: Positive
- 107 Internationalism: Positive
- 108 European Integration: Positive
- 109 Internationalism: Negative
- 110 European Integration: Negative

**Domain 2: Freedom and Democracy**
- 201 Freedom and Human Rights: Positive
- 202 Democracy: Positive
- 203 Constitutionalism: Positive
- 204 Constitutionalism: Negative

**Domain 3: Political System**
- 301 Decentralisation: Positive
- 302 Centralisation: Positive
- 303 Governmental and Administrative Efficiency: Positive
- 304 Political Corruption: Negative
- 305 Political Authority: Positive

**Domain 4: Economy**
- 401 Free Enterprise: Positive
- 402 Incentives: Positive
- 403 Market Regulation: Positive
- 404 Economic Planning: Positive
- 405 Corporatism: Positive
- 406 Protectionism: Positive
- 407 Protectionism: Negative
- 408 Economic Goals
- 409 Keynesian Demand Management: Positive
- 410 Economic Growth
- 411 Technology and Infrastructure: Positive
- 412 Controlled Economy: Positive
- 413 Nationalisation: Positive
- 414 Economic Orthodoxy: Positive
- 415 Marxist Analysis: Positive
- 416 Anti-Growth Economy: Positive

**Domain 5: Welfare and Quality of Life**
- 501 Environmental Protection: Positive
- 502 Culture: Positive
- 503 Equality: Positive
- 504 Welfare State Expansion
- 505 Welfare State Limitation
- 506 Education Expansion
- 507 Education Limitation

**Domain 6: Fabric of Society**
- 601 National Way of Life: Positive
- 602 National Way of Life: Negative
- 603 Traditional Morality: Positive
- 604 Traditional Morality: Negative
- 605 Law and Order: Positive
- 606 Civic Mindedness: Positive
- 607 Multiculturalism: Positive
- 608 Multiculturalism: Negative

**Domain 7: Social Groups**
- 701 Labour Groups: Positive
- 702 Labour Groups: Negative
- 703 Agriculture: Positive
- 704 Middle Class and Professional Groups: Positive
- 705 Minority Groups: Positive
- 706 Non-Economic Demographic Groups: Positive
- 000 No meaningful category applies
Two-step process

e.g.: "We need to address our close ties with our neighbours (107) / as well as the unique challenges facing small business owner in this of economic hardship (402)"

https://manifestoproject.wzb.eu/coding_schemes/mp_v5

Since 2015 some minor changes to the categories have been implemented, see:

https://manifestoproject.wzb.eu/down/papers/Evolution_of_the_Manifesto_Coding_Instructions_and_the_Category_Scheme.pdf
Two-step process

Note that the number you find in the CMP dataset in correspondence to each category is **not the ‘raw’** numbers of quasi-sentences coded into each category, but the **percentage** calculated out of the total number of references to all categories.

For example, if you find the number 3.2 in correspondence to the per107 category, this means that 3.2% of all the quasi-sentences of that party in that manifesto is devoted to discuss the per107 category (*Internationalism: Positive*). Why doing that?

Through this “standardization” it becomes possible to directly compare electoral programs irrespective of their relative length (short or long)!!!
The CMP actual coding scheme explicitly refers to a particular “saliency” theory of politics.

What’s that? We do not have time to discuss about it, but see the Appendix file to this lecture on the home-page of the course!
CMP and the Japanese case

Until very recently, CMP researchers used so-called *kōyaku* (‘public promises’) instead of Japanese party manifests. These electoral pledges are short answers extracted from party leaders in rapid-fire pre-election interviews by a national daily newspaper.

Because they are short, and because the selection of topics was truncated and selected by the interviewers, these text statements are different from election manifests, in which parties can freely choose to emphasize any issues they want.

Consequently, Japan’s ‘manifestos’ have the second shortest average text length in the CMP data set (74 sentences on average), and one of the highest proportions of CMP policy categories with zero entries (69% on average).
Manifesto data can be used, and have been used, to provide valid and reliable measurements of party policy position. You can estimate parties’ positions over several policy dimensions. However, the most used one using CMP data is the Left-right scale estimates (Budge et al. 2001). But how to do that? Two different ways: a-priori (i.e., supervised) and a-posteriori (i.e., unsupervised).
CMP’s left-right scale

A-priori approaches

13 leftist categories vs. 13 rightist categories
# CMP’s left-right

<table>
<thead>
<tr>
<th>Left categories</th>
<th>Right categories</th>
</tr>
</thead>
<tbody>
<tr>
<td>103 Anti-imperialism</td>
<td>104 Military: positive</td>
</tr>
<tr>
<td>105 Military: negative</td>
<td>201 Freedom and human rights</td>
</tr>
<tr>
<td>106 Peace</td>
<td>203 Constitutionalism: positive</td>
</tr>
<tr>
<td>107 Internationalism: positive</td>
<td>305 Political authority</td>
</tr>
<tr>
<td>202 Democracy</td>
<td>401 Free enterprise</td>
</tr>
<tr>
<td>403 Market regulation</td>
<td>402 Incentives</td>
</tr>
<tr>
<td>404 Economic planning</td>
<td>407 Protectionism: negative</td>
</tr>
<tr>
<td>406 Protectionism: positive</td>
<td>414 Economic orthodoxy</td>
</tr>
<tr>
<td>412 Controlled economy</td>
<td>505 Welfare state limitation</td>
</tr>
<tr>
<td>413 Nationalisation</td>
<td>601 National way of life: positive</td>
</tr>
<tr>
<td>504 Welfare state expansion</td>
<td>603 Traditional morality: positive</td>
</tr>
<tr>
<td>506 Education expansion</td>
<td>605 Law and order</td>
</tr>
<tr>
<td>701 Labour groups: positive</td>
<td>606 Social harmony</td>
</tr>
</tbody>
</table>

Source: Budge et al. (2001), Mapping Policy Preferences, Appendix III. Left-right score = proportion (right − left) × 100.
The basis of the selection of such 26 categories is mainly theoretical: highly influential early modern theorists put them together in their political analyses.

**Left categories** are linked in Marxist and progressive political analyses of around 1900, and contrast with **Right topics** linked together in opposing analyses broadly supporting the existing order and market.

Having said that, these categories tend also to empirically co-vary (according to factor-analysis).
CMP left-right measure: the RILE measure

But how to move from these 26 categories to a left-right score?

First possibility: the RILE measure

Each party’s position is measured as the difference (in percentages) between the sum of the right-associated text mentions (R) and the sum of the left-associated ones (L):

\[ RILE = (R\% - L\%) \]

The final scale ranges from -100 to 100

This is the official CMP left-right measure
CMP left-right measure: the RILE measure

In terms of ‘raw’ numbers of quasi-sentences coded into each category (rather than their percentage) we can write:

\[ \text{RILE (raw scores)} = \frac{R - L}{R + L + O} = \frac{R - L}{N} \]

Where \( O \) are the “other categories”

This measure is based on the difference in counts between left and right sentences counts normalized by the total number of sentences \( N \) in the manifesto on any issue.

From this definition it is clear that each count in \( L \) or \( R \) has the same fixed marginal effect: \( 1/N \)
An application of the RILE measure to USA
An application of the RILE measure to USA

Which trends do you see?

1) The largest difference ever among parties nowadays

2) Gop pretty stable since Reagan. The Dems moving very much to the left (the heritage of Bill Cling has gone?)

3) And overall, what about the party system? Stable?
An application of the RILE measure to USA

Evolution of USA Left-Right Scale (black line: weighted mean)

Parties

Democratic Party  Republican Party

Elections (DATE)

Left-Right (RILE)
One possible problem with the RILE measure:
Imagine two situations: 1) a 200-sentence manifesto with 100 right sentences and no left sentences; 2) the same manifesto with 50 sentences added that are neither left nor right (e.g. on the environment)
CMP left-right measure: the RILE measure

The RILE score would change from \((50 - 0) = 50\) to its RILE score to \((40 - 0) = 40\), suggesting that the party shifted 20% toward the left.

The party is thus scored as less Right-leaning in the second election compared to the first even though the proportion of left and right sentences, the raw material for expressing a position, have not changed…

…that is, it has moved towards the centre by virtue of devoting more attention to topics that are not purely Left or Right.

For the RILE scale, this means that counts of the categories not in the scale still affect estimated party positions.
Underlying assumption: if an issue becomes less important then a party will devote fewer sentences to it!

This can be justified on the grounds that manifests are not just a compilation of discrete policy stands but an integrated and complete statement all of whose constituent parts have been carefully considered in relation to each other by programs committees and party conventions and approved as a whole by the latter
CMP left-right measure: the RATIO scale

To remedy this bias a ‘ratio’ scale has been proposed where only ‘Left’ and ‘Right’ emphases are used in the denominator, as follows:

\[ \text{RATIO (raw scores or percentage is the same in this case)} = \frac{R - L}{R + L} \text{ or } \frac{R\% - L\%}{R\% + L\%} \]
CMP left-right measure: the RATIO scale

The measure ranges from -1 to 1: dividing by R + L decouples the measure from variation in the importance a party assigns to any issue area (other than R and L).

Unlike RILE, this measure will not necessarily create an apparent move to a more centrist position if the party decides to focus on other policy areas: in the previous example a party would get always a score of +1!

Indeed, comparing category counts only to counts in the opposing category rather than to counts of all quasi-sentences, makes the marginal effect of another sentence on the left or right side of the issue equals to: 1/(R + L) (but still with a fixed marginal effect!)
CMP left-right measure: the RATIO scale

Some problems...

RATIO shares the assumptions embodied in RILE about the existence of fixed endpoints (in this case -1 and +1 rather than -100 and +100 as in RILE)

This has the unfortunate effect of forcing scores to the extremes, i.e., forcing the RATIO to -1 when R = 0 irrespective of the value of L, or to 1 when L = 0 irrespective of the value of R, leading to spikes at the boundaries of the scale
CMP left-right measure: the LOGIT scale

Consider the process of reading a party manifesto for changes in policy content, as a voter might do, for example, if trying to identify any change in some party’s policy position on the European Union.

If the party’s previous platform contained 50 sentences in favor of increased European integration, and 20 emphasizing its disadvantages, then a new manifesto containing 50 sentences in favor and 21 against would barely register as an indicator of policy change.

But if the previous platform had contained 10 and 4 sentences for and against the EU, and the new platform 10 and 5, then a policy change would be perceived as more plausible.
This suggests that from the point of view of a party manifesto writer wanting to communicate a position effectively, it is important to manipulate the relative balance between quantity of sentences, i.e., the ratio (rather than the difference) between sentence counts (in favor of the EU and against it going back to our previous example or R/L if we go back to the left-right scale).
Following this logic, the effect of adding one more sentence in the first case decreases the ratio of anti-EU sentences by about 5% \((50/20=2.5; \frac{50}{21}=2.38; (2.38/2.5)-1=-5\%)\), and in the second by 20% \((10/4)=2.5; \frac{10}{5}=2; (2/2.5)-1=-20\%)\).
The marginal effect of one more sentence is now decreasing in the amount that has already been said on the topic. This is supported by evidence from psychology: there is a decreasing marginal effect of an extra unit. The size of the “perceivable difference” of a subjective quantity is a constant proportion of the quantity already present.
Still working with R/L is not always that easy… Imagine that a party has R=0 and L=1, or R=0 and L=100, if you use a proportion (R/L) you would always gets 0. Is it reasonable? And if L=0 and R=1 (or 100)? How to deal with these problems?
Use a logarithmic scale relationship!

\[
\text{LOGIT (raw scores)} = \log\left(\frac{R+.5}{L+.5}\right)
\]

that becomes…

\[
\text{LOGIT (raw scores)} = \log(R+.5) - \log(L+.5) \quad [1]
\]

Why a logarithmic scale? Imagine that a party has \(R=0\) and \(L=1\), or \(R=0\) and \(L=100\), if you do not use the logarithmic scale but just a proportion \((R/L)\) you would always gets 0, while using [1] you would get -0.47 and -2.3. Much more reasonable!

And if \(L=0\) and \(R=1\) (or 100)? You could not estimate \(R/L\), but [1] yes!
CMP left-right measure: the LOGIT scale

But why adding 0.5 (that we call “offset”) to all counts? This smooths LOGIT scores slightly towards 0 and makes position estimates created from very small counts more stable, while barely affecting those derived from more reasonable numbers of sentences.
Like RATIO, LOGIT is **conditional** because it only considers sentences that are assigned to left or right. Unlike RILE and RATIO, however, LOGIT has **no predefined end points**: given enough sentences, it is possible to generate positions of any level of extremity. However, although any real valued policy position can be represented, expressing **extreme positions** requires **exponentially more sentences** in L or R to move the policy position the same distance left or right.
CMP left-right measure: the LOGIT scale

We can compute the LOGIT directly on the percentages reported in the CMP dataset. This has no effect on the ratio $R/L$, although it does affect the offset (i.e., the 0.5 in [1]).

Using the percentages, we must therefore rescale the offset $= 100*(0.5/N)$ according to the number of total quasi-sentences in a given program so that it can be directly comparable to $L\% = 100*(L/N)$ (the sum of the % belonging to the left categories) and $R\% = 100*(R/N)$.
CMP left-right measure: the LOGIT scale

To take an example, consider the case of a party with $L\% = 3.7$ and $R\% = 0$ and total number of codified sentences = 81

$N = 81$
$L = (3.7/100) \times 81 = 3$
$R = (0/100) \times 81 = 0$
$L\% = 3.7$
$R\% = 0$

offset = 0.5
offest_new = 100 \times (0.5/81) = 0.617$

LOGIT (traditional) = $\log(R + 0.5) - \log(L + 0.5) = \log(0 + .5) - \log(3 + .5)$
$= \log(.5) - \log(3.5) = -1.945$

LOGIT (with percentage) = $\log(R\% + 0.617) - \log(L\% + 0.617)$
$= \log(0.617) - \log(3.7 + 0.617) = -1.945$
Possible problems? Much ado about nothing!
The LOGIT procedure correlates with the original RILE ones at $r = .94$
Evolution of USA Left-Right Scale: Cons vs Lab

Parties
Democratic Party
Republican Party
CMP left-right measure: the vanilla method

*Posteriori* approach:

Gabel and Huber (2000) approach: ideology is conceived as a "constraint" on policy positions, such that positions on a broad range of issues are related to each other in consistent and identifiable ways.

Ideology therefore reduces differences in party positions over many policies to differences in party positions on a single dimension.
CMP left-right measure: the vanilla method

Empirically, the **left-right dimension** is then defined inductively as the **super issue** that most constrains parties' positions across a broad range of policies.

The *vanilla method* aims to uncover this "super issue" and party positions on it.

Specifically this method involves extracting one dimension from the CMP data using a **factor analysis (FA)** with the attempt to identify the **underlying single dimension** that best account for the observed co-variation from all the policy categories across parties.

What is a FA? An example
Data Reduction

Do we really need two dimensions to capture all the variance in party positions?
Data Reduction

Better using one dimension (and we do not lose so much information after all...)

Social policy

Economy
Data Reduction

FA is used to **reduce the number of dimensions** within a data set **by choosing** only those “factors” (1 or more) that account for **most of the variation in the original multivariate data** and to summarize the data with little loss of information by projecting them onto a lower dimensional subspace. This can result in a **good approximation** of the original data.

FA is especially useful when there is a **high-degree of correlation** present in our dataset (if correlation in your data is zero, no way to do any reasonable dimension reduction!!!)
CMP left-right measure: the vanilla method

In the case of Gabel and Huber, they want to reduce the original 56 number of dimensions presented in CMP by focusing on one single latent dimension that best accounts the overall correlation among those (56) dimensions.

Note that contrary to RILE (and other methods) wherein all categories are given equal weight in defining the left-right dimension, the VANILLA method used in identifying the underlying dimension assigns weights to the different categories based on the extent of their covariation with other categories.
CMP left-right measure: the vanilla method

We can apply this approach under a variety of assumptions about how **ideology varies across time and space**

It may be that the meaning of left-right ideology differs **across nations but is stable within nations over time**.

Based on this assumption, on should estimate the left-right dimension using which manifestos?

Manifestos data pooled by country
CMP left-right measure: the vanilla method

Alternatively one could assume that the meaning of left-right may be **common across nations but differ over time**. Based on this assumption, one should estimate the left-right dimension using...

....manifestos data for different time periods but pooled across nations

A third possibility is that the substantive issues underlying left-right ideology have been roughly the **same across countries and over time** since WWII (like does RILE). In this case, one should estimate the left-right dimension with pooled manifestos data from all relevant countries in all available years. Gabel and Huber suggest to implement their method **under this specific scenario**!
CMP left-right measure: the vanilla method

VANILLA’ scaling problems

The essential purpose of the Gabel-Huber method is to use the CMP data to generate the best uni-dimensional account of covariation in party policy positions.

While there is no a-posteriori interpretation of results using the Gabel-Huber approach, the price paid is that the notions of left and right that emerge have no substantive meaning in terms of public policy.
CMP left-right measure: the vanilla method

VANILLA’ scaling problems

Moreover, any give implementation of this inductive technique will be entirely dependent upon the choice of cases to analyse. Applying this method to different parties and different time periods will produce left-right scales with different substantive meanings. Besides, the scores keep changing the more data you have!

On the contrary, estimating the left-right in any of the a-priori methods discussed earlier produces a comparative and over time invariant measure.
Human-coded content analysis is a painfully resource-intensive activity. Given the huge expense involved, the vast majority of the manifestos that form the basis of the CMP dataset were coded once only by a single human coder. A crucial consequence of this is that every single number in the CMP dataset, as in almost all other datasets generated by human-coded content analysis, is presented as a single point estimate with no estimate of associated error. But there is surely an error in these, as in all other, data.
CMP’s data shortcomings

The crucial implication of having no estimate of associated error is that, when evaluating the difference between the estimated positions of two parties (or the same party at two points in time), we have no way of knowing, systematically, whether these positions are “the same” or “different.”

The same two positions might be judged to be the same if they had large standard errors, or different if their standard errors were small, that is…

…it is impossible, in the CMP data, to distinguish measurement error from “real” underlying change in the policy positions under investigation.
CMP’s data shortcomings

How to deal with that?

We can **simulate the error-generating processes** by which CMP data are generated! But how to do that?

If you are interested, take a look at here:


…and if you have questions, including how to estimate bootstrap s.e. in R with respect to CMP data, just contact me!
Before today’s lab class

1. `install.packages("manifestoR", repos='http://cran.us.r-project.org')`
2. `install.packages("psych", repos='http://cran.us.r-project.org')`
3. `install.packages("PerformanceAnalytics", repos='http://cran.us.r-project.org')`
4. `install.packages("corrplot", repos='http://cran.us.r-project.org')`
5. `install.packages("corrgram", repos='http://cran.us.r-project.org')`
6. `install.packages("latticeExtra", repos='http://cran.us.r-project.org')`
7. `install.packages("DT", repos='http://cran.us.r-project.org')`
IMPORTANT!!!
Before using rtweet & streamR

1. You need to have a twitter account. if you do not have one go to [http://twitter.com/signup](http://twitter.com/signup) and set one up. Also you need to have a mobile number as part of this account
Before using rtweet & streamR

2. Now that you have created a twitter account you need to go to https://apps.twitter.com and sign on with your twitter account.
How to use rtweet & streamR

3. Once you have signed in, you should see the following screen, and simply click on the button that says “Create New App”.

![Create New App Button](image-url)
Before using rtweet & streamR

4. Once you click on the “Create New App” button you will go to the Create an Application screen. There are three fields, a click box and a button you need to click on this page.

Create an application

Application Details

Name *
Your application name. This is used to attribute the source of a tweet and in user-facing authorization screens. 32 characters max.

Description *
Your application description, which will be shown in user-facing authorization screens. Between 10 and 200 characters max.

Website *
Your application’s publicly accessible homepage, where users can go to download, make use of, or find out more information about your application. This fully-qualified URL is used in the source attribution for tweets created by your application and will be shown in user-facing authorization screens. (If you don’t have a URL yet, just put a placeholder here but remember to change it later)

Callback URL
Where should we return after successfully authenticating? OAuth 1.0a applications should explicitly specify their oauth_callback URL on the request token step, regardless of the value given here. To restrict your application from using callbacks, leave this field blank.

Developer Agreement

Effective: May 18, 2015.
Before using rtweet & streamR

The three fields are Name, Description and Website. The name of the application must be unique so this may take a few tries. The description needs to be at least 10 character long, and put in a website.

If you do not have one you can use https://bigcomputing.blogspot.com
Before using rtweet & streamR

5. Then follow the instructions provided here:
https://rtweet.info/articles/auth.html
Before using rtweet & streamR

6. Once you have successfully created an application and ended all the procedures, you should have your own:

A. Consumer Key (API Key)
B. Consumer Secret (API Secret)
C. Access Token
D. Access Token Secret
Before using rtweet & streamR

For rtweet
- To run a R session you must enter the following code using the previous four pieces of information:

```r
install.packages("rtweet", repos='http://cran.us.r-project.org')
library(rtweet)
token <- create_token(
  app = "my_twitter_research_app",
  consumer_key = "your_consumer_key" [that you got in the previous step!],
  consumer_secret = "your_consumer_secret" [that you got in the previous step!],
  access_token = "your_access_token" [that you got in the previous step!],
  access_secret = "your_access_secret" [that you got in the previous step!]
)
get_token()
## check to see if the token is loaded
identical(token, get_token())
```
Before using rtweet & streamR

For rtweet

- To check that everything worked fine, then type the following command (you should get some tweets back!):

```r
# Search for up to 100 (non-retweeted) tweets containing the rstats hashtag
rt <- search_tweets("#rstats", n = 100, include_rts = FALSE)
head(rt, n = 3)
```
Before using rtweet & streamR

For streamR: the first time you use streamR enter the following code:

```r
install.packages("streamR", repos='http://cran.us.r-project.org')
install.packages("ROAuth", repos='http://cran.us.r-project.org')
library(streamR)
library(ROAuth)
# download this file in your usual working directory via setwd
download.file(url="http://curl.haxx.se/ca/cacert.pem", destfile="cacert.pem")
requestURL <- "https://api.twitter.com/oauth/request_token"
accessURL <- "https://api.twitter.com/oauth/access_token"
authURL <- "https://api.twitter.com/oauth/authorize"
consumerKey <- "your consumer key"
consumerSecret <- "your consumer secret"
my_oauth <- OAuthFactory$new(consumerKey = consumerKey, consumerSecret = consumerSecret, requestURL = requestURL, accessURL = accessURL, authURL = authURL)
```
Before using rtweet & streamR

For streamR:

# then, after typing the following line, your browser will be automatically launched:
my_oauth$handshake(cainfo = system.file("CurlSSL", "cacert.pem", package = "RCurl"))
Before using rtweet & streamR

For streamR:

# after clicking on “Authorize app” a new display will appear

You've granted access to TwitterLuigiApp!

Next, return to TwitterLuigiApp and enter this PIN to complete the authorization process:

8 2

# paste and copy the PIN back in R

```r
> my_oauth$handshake(cainfo = system.file("CurlSSL", "cacert.pem", package = "R")
To enable the connection, please direct your web browser to:
https://api.twitter.com/oauth/authorize?oauth_token=muUnzAAAAAAAPllyGRAAABYDQ1hw8
When complete, record the PIN given to you and provide it here.
```
Before using rtweet & streamR

For streamR:

Then run this line to save your procedure:

```
save(my_oauth, file = "my_oauth.Rdata")
```

After this very first time, all you just to do everytime you want to run a session in R is typing the following line by calling the file that with all the procedure you have just saved:

```
load("my_oauth.Rdata")
```
Before using rtweet & streamR

For streamR

- To check that everything worked fine, then type the following command (you should get some tweets back after 15 seconds!):

```r
filterStream("tweetsTrump.json", track = c("Trump"), timeout = 15,
    oauth = my_oauth)
tweets.df <- parseTweets("tweetsTrump.json")
str(tweets.df)
```