Damegender Manual: Counting Males and Females in Internet Communities

for version 0.2.12, 20 Sep 2020

David Arroyo Menéndez (`davidam@gnu.org`)

This manual is for Damegender (version 0.2.12, 20 Sep 2020).

The sources will be find in https://github.com/davidam/damegender/tree/master/manual

Table of Contents

1 Introduction

Damegender is a gender detection tool from the name coded by David Arroyo MEnéndez (DAME).

The gender detection tools from the names are being used usually with commercial APIs. But many countries has been doing efforts in the last years for contribute names and a number of people using each name with Open Data Licenses. So, this software is collecting this effort on an original way (we are using Machine Learning algorithms for predict names that is not appearing in our database).

Damegender is giving measures to compare in any moment our solution with the commercial APIs. So, the user can understand when it's useful to invest money or not depending of the dataset. Damegender allows to the users download a big number of names from a csv file.

This software is written oriented to tests. So you can check the right behaivour of the software with python tests for the classes and methods and with shell tests for the python commands.

Damegender is using Perceval for count males and females in a lot of Internet Communities (wikis, mailing lists, software repositories, bug tracking systems, ...). We shows source for count males and females in different situations (Ex: count-debian-gender.py)

This software is taking into account the power to predict nations and ethnicity from the surnames (Ex: surname.py, surnameincountries.py and ethnicity.py).

2 Installation

Possible Debian/Ubuntu dependencies:

```
$ sudo apt-get install python3-nose-exclude python3-dev dict dict-freedict-eng-spa dict-freedict-spa-eng dictd
```

Now, to install damegender from sources:

```
$ git clone https://github.com/davidam/damegender
$ cd damegender
$ pip3 install -r requirements.txt
```

Now, to install damegender with python package:

```
$ python3 -m venv /tmp/d
$ cd /tmp/d
$ source bin/activate
$ pip install --upgrade pip
$ pip3 install damegender
$ cd lib/python3.5/site-packages/damegender
$ python3 main.py David
```

To install apis extra dependencies:

```
$ pip3 install damegender[apis]
```

To install mailing lists and repositories extra dependencies:

```
$ pip3 install damegender[mails_and_repositories]
```

To install all possible dependencies

```
$ pip3 install damegender[all]
```

Currently you can need an api key from:

- https://store.genderize.io/documentation
- https://gender-api.com
- https://www.nameapi.org/
- https://v2.namsor.com/NamSorAPIv2/sign-in.html

To configure your api key you can execute:

```
$ python3 apikeyadd.py
```

3 Commands

You must start to check tests to understand that all is ok:

```
$ cd src/damegender
$ ./testsbycommands.sh         # It must run for you
$ ./testsbycommandsextralocal.sh  # You will need all dependencies
                                  # with: $ pip3 install damegender[all]
$ ./testsbycommandsextranet.sh    # You will need api keys
```

You can continue check python tests:

Execute all tests:

```
$ nosetests3 tests
```

Execute one file:

```
$ nosetests3 tests/test_basics.py
```

Execute one test:

```
$ nosetests3 tests/test_basics.py:TestBasics.test_indexing
```

If you are in a fresh installation, perhaps you want regenerate by your own risk some files downloaded to understand how it has been generated:

```
$ python3 postinstall.py
```

You can find an big list of commands to execute this shell scripts. Now a detailed execution of some selected examples:

The first command to learn is main.py. You can play now with this command:

```
# Detect gender from a name (INE is the dataset used by default)
$ python3 main.py David
David gender is male
363559  males for David from INE.es
0 females for David from INE.es

# Detect gender from a name only using machine learning (experimental way)
$ python3 main.py Agua --ml=nltk
Agua gender is female
0 males for Agua from INE.es
0 females for Agua from INE.es

# Detect gender from a name (all census and machine learning)
$ python3 main.py David --verbose
365196 males for David from INE.es
0 females for David from INE.es
1193 males for David from Uruguay census
5 females for David from Uruguay census
26645 males for David from United Kingdom census
0 females for David from United Kingdom census
3552580 males for David from United States of America census
12826 females for David from United States of America census
David gender predicted with nltk is male
David gender predicted with sgd is male
David gender predicted with svc is male
David gender predicted with gaussianNB is male
David gender predicted with multinomialNB is male
David gender predicted with bernoulliNB is male
David gender predicted with forest is male
David gender predicted with tree is male
David gender predicted with mlp is male
```

The first Free Software for gender detection tool was created in C language program and you can look for a python version with the name genderguesser. Some people was working in a Free dataset called name_dict.txt with 48500 names. I want to give thanks to this effort with nameincountries.py due to the good work organizing many names in different countries.

```
$ python3 nameincountries.py David
grep -i " David " files/names/nam_dict.txt > files/grep.tmp
males: ['Albania', 'Armenia', 'Austria', 'Azerbaijan', 'Belgium', 'Bosnia and Herze-
govina', 'Czech Republic', 'Denmark', 'East Frisia', 'France', 'Georgia', 'Ger-
many', 'Great Britain', 'Iceland', 'Ireland', 'Israel', 'Italy', 'Kaza-
khstan/Uzbekistan', 'Luxembourg', 'Malta', 'Norway', 'Portugal', 'Roma-
nia', 'Slovenia', 'Spain', 'Sweden', 'Swiss', 'The Netherlands', 'USA', 'Ukraine']
females: []
both: []
```

This Free Software has been developed in the frame of a Phd in the Universidad Rey Juan Carlos I with the Phd director Jesús González Barahona, so I have developed some commands to use Perceval (Free Software where he has done good contributions)

To count gender from a git repository:

```
$ python3 git2gender.py https://github.com/chaoss/grimoirelab-perceval.git -
-directory="/tmp/clonedir"
The number of males sending commits is 15
The number of females sending commits is 7
```

You can see a verbose output using the spanish dataset (–language=es) for males and females with:

```
$ python3 git2gender.py https://git.drupalcode.org/project/orgmode.git -
-directory=/tmp/orgmode --show=all --verbose --language=es
You are not using ml the process is not very slow, but perhaps you are not find-
ing good results
The number of males sending commits is 2
The list of males sending commits is:
['David Arroyo Menendez', 'David Arroyo']
David Arroyo Menéndez <davidam@es.gnu.org> (67 commits)
David Arroyo Menendez <davidam9@riseup.net> (49 commits)
David Arroyo Menéndez <davidam@gmail.com> (20 commits)
David Arroyo Menendez <david.arroyo@edoctores.com> (10 commits)
David Arroyo Menendez <davidam@es.gnu.org> (14 commits)
David Arroyo7 <davidam@es.gnu.org> (13 commits)
David Arroyo7 <davidam@gnu.org> (10 commits)
The number of females sending commits is 1
The list of females sending commits is:
['Mireia Lopez']
Mireia López <mireia@omada.es> (23 commits)
The number of people with unknown gender sending commits is 0
The list of people with unknown gender sending commits is:
[]
```

To count gender from a mailing list:

```
$ cd files/mbox
$ wget -c http://mail-archives.apache.org/mod_mbox/httpd-announce/201706.mbox
$ cd ../..
$ python3 mail2gender.py http://mail-archives.apache.org/mod_mbox/httpd-
announce/
You are not using ml the process is not very slow, but perhaps you are not find-
ing good results
The number of males sending mails is 24
The number of females sending mails is 2
The number of people with unknown gender sending mails is 5
```

You can execute a verbose output with:

```
$ python3 mail2gender.py http://mail-archives.apache.org/mod_mbox/httpd-
announce/ --verbose --show=all
You are not using ml the process is not very slow, but perhaps you are not find-
ing good results
The number of males sending mails is 24
The list of males sending mails is:
['Jim Jagielski <jim@apache.org>', 'Jacob Champion <jchampion@apache.org>', 'DEN-
NIS BIN XAVIER <balaranpillai@gmail.com>', '"Leonard Lausen (Jira)" <jira@apache.org>'
ing (Jira)" <jira@apache.org>', '"Roman Shaposhnik (Jira)" <jira@apache.org>', '"Bertr
retaz (Jira)" <jira@apache.org>', '"Mark Thomas (Jira)" <jira@apache.org>', '"Justin M
mon Phipps (Jira)" <jira@apache.org>', '"Chris Olivier (Jira)" <jira@apache.org>', 'Ja
hoda <lahoda@gmail.com>', '"Michael Vorburger (Jira)" <jira@apache.org>', '"Ralph Go-
ers (Jira)" <jira@apache.org>', '"Jens Geyer" <jensg@apache.org>', 'Mark Thomas <markt
curu (Jira)" <jira@apache.org>', '"Kevin A. McGrail (Jira)" <jira@apache.org>', '"Gor-
don (Jira)" <jira@apache.org>', 'Gary Gregory <garydgregory@gmail.com>', '"Owen O\'Mal
The number of females sending mails is 2
The list of females sending mails is:
['Riya Singh <hellen.serviceweb@hotmail.com>', '"Hannah Zacharski (Jira)" <jira@apache
The number of people with unknown gender sending mails is 5
The list of people with unknown gender sending mails is
['"SimpaticoTech.it OFFERTE" <web.info@simpaticotech.it>', 'Simpatico Net-
work srl <web.info@simpaticotech.it>', 'gmcdonald@apache.org', 'Hen <ba-
yard@apache.org>', '"Jean-Baptiste Briaud (Jira)" <jira@apache.org>']
```

Perhaps you don't know a name, but you have obtained an free key for an api to retrieve it:

```
$ python3 api2gender.py Leticia --surname="Martin" --api=namsor
```

```
female
scale: 0.99
```

If you want to know the gender of a good number of names you can download results from an api and save in a file with downloadjson.py

```
$ python3 downloadjson.py --csv=files/names/min.csv --api=genderize
$ cat files/names/genderizefiles_names_min.csv.json
```

Now we are going to learn some commands for measure the successful of our solution:

```
$ python3 accuracy.py --csv=files/names/min.csv
#################### NLTK!!
Gender list: [1, 1, 1, 1, 2, 1, 0, 0]
Guess list:  [1, 1, 1, 1, 0, 1, 0, 0]
Dame Gender accuracy: 0.875
```

```
$ python3 confusion.py --csv="files/names/partial.csv" --api=nameapi --
jsondownloaded="files/names/nameapifiles_names_partial.csv.json"
A confusion matrix C is such that Ci,j is equal to the number of obser-
vations known to be in group i but predicted to be in group j.
If the classifier is nice, the diagonal is high because there are true positives
Nameapi confusion matrix:

[[ 3, 0, 0]
 [ 0, 15, 1]]
```

```
$ python3 errors.py --csv="files/names/all.csv" --api="genderguesser"
Gender Guesser with files/names/all.csv has:
+ The error code: 0.22564457518601835
+ The error code without na: 0.026539047204698716
+ The na coded: 0.20453365634192766
+ The error gender bias: 0.0026103980857080703
```

You can generate a lot of logs about errors, accuracies and/or confusion:

```
$ ./logs-accuracies.sh
$ ./logs-confusion.sh
$ ./logs-errors.sh
```

Perhaps you are interested on reproduce experiments to determine features:

```
$ python3 infofeatures.py
Females with last letter a: 0.4705246078961601
Males with last letter a: 0.048672566371681415
Females with last letter consonant: 0.2735841767750908
Males with last letter consonant: 0.6355328972681801
Females with last letter vocal: 0.7262612995441552
Males with last letter vocal: 0.3640823393612928
$ python3 pca-components.py --csv="files/features_list.csv" # To deter-
mine number of components
$ python3 pca-features.py                                    # To under-
stand the weight between variables for a target
```

Now we can go to play with surnames:

```
$ python3 surname.py Gil --total=es
```

```
There are 140004 people using Gil in Spain

$ python3 surname.py Lenon --total=us
There are 837 people using Lenon in United States of America

$ python3 ethnicity.py Smith
In United States of America the percentages about the race of Smith surname is:
White: 73.35
Black: 22.22
Hispanic: 1.56
Asian Pacific Indian American: 0.40
American Indian and Alaska Native: 0.85
Various races: 1.63
```

4 Statistics

In the last chapter we were learning to execute some commands such as accuracy.py, confusion.py, or errors.py, but perhaps you need to understand more theory about statistics to understand why this commands is being interesting for you.

4.1 Measuring success and error

To guess the sex, we have an true idea (example: female) and we obtain a result with a method (example: using an api, querying a dataset or with a machine learning model). The guessed result could be male, female or perhaps unknown. Remember some definitions about results about this matter:

True positive is to find a value guessed as true if the value in the data source is positive.

True negative is to find a value guessed as true if the the value in the data source is negative.

False positive is to find a value guessed as false if the the value in the data source is positive.

False negative is to find a value guessed as false if the the value in the data source is negative.

So, we can find a vocabulary for measure true, false, success and errors. We can make a summary in the gender name context about mathematical concepts:

Precision is about true positives divided by true positives plus false positives

```
(femalefemale + malemale ) /
(femalefemale + malemale + femalemale)
```

Recall is about true positives divided by true positives plus false negatives.

```
(femalefemale + malemale ) /
(femalefemale + malemale + malefemale + femaleundefined + maleundefined)
```

Accuray is about true positives divided by all.

```
(femalefemale + malemale ) /
(femalefemale + malemale + malefemale + femalemale + femaleundefined + maleundefined)
```

The F1 score is the harmonic mean of precision and recall taking both metrics into account in the following equation:

```
2 * (
(precision * recall) /
(precision + recall))
```

In Damengender, we are using accuracy.py to apply these concepts. Take a look to practice:

```
$ python3 accuracy.py --api="damegender" --measure="f1score" --csv="files/names/partia
-jsondownloaded=files/names/partialnoundefined.csv.json
################### Damegender!!
Gender list: [1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0,
Guess list:  [1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0,
Damegender f1score: 0.9090909090909091
```

```
$ python3 accuracy.py --api="damegender" --measure="recall" --csv="files/names/partial
-jsondownloaded=files/names/partialnoundefined.csv.json
################### Damegender!!
Gender list: [1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0,
Guess list:  [1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0,
Damegender recall: 1.0

$ python3 accuracy.py --api="damegender" --measure="accuracy" --csv="files/names/parti
-jsondownloaded=files/names/partialnoundefined.csv.json
################### Damegender!!
Gender list: [1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0,
Guess list:  [1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0,
Damegender accuracy: 0.8571428571428571

$ python3 accuracy.py --api="genderguesser" --measure="accuracy" --csv="files/names/pa
-jsondownloaded=files/names/partialnoundefined.csv.json
################### Genderguesser!!
Gender list: [1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0,
Guess list:  [1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0,
Genderguesser accuracy: 0.8571428571428571

$ python3 accuracy.py --api="genderguesser" --measure="precision" --csv="files/names/p
-jsondownloaded=files/names/partialnoundefined.csv.json
################### Genderguesser!!
Gender list: [1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0,
Guess list:  [1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0,
Genderguesser precision: 0.9090909090909091

$ python3 accuracy.py --api="genderguesser" --measure="recall" --csv="files/names/part
-jsondownloaded=files/names/partialnoundefined.csv.json
################### Genderguesser!!
Gender list: [1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0,
Guess list:  [1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0,
Genderguesser recall: 1.0

$ python3 accuracy.py --api="genderguesser" --measure="f1score" --csv="files/names/par
-jsondownloaded=files/names/partialnoundefined.csv.json
################### Genderguesser!!
Gender list: [1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0,
Guess list:  [1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0,
Genderguesser f1score: 0.9090909090909091
```

Error coded is about the true is different than the guessed:

```
(femalemale + malefemale + maleundefined + femaleundefined) /
(malemale + femalemale + malefemale +
femalefemale + maleundefined + femaleundefined)
```

Error coded without na is about the true is different than the guessed, but without undefined results.

```
(maleundefined + femaleundefined) /
(malemale + femalemale + malefemale +
femalefemale + maleundefined + femaleundefined)
```

Error gender bias is to understand if the error is bigger guessing males than females or viceversa.

The **weighted error** is about the true is different than the guessed, but giving a weight to the guessed as undefined.

```
(femalemale + malefemale +
+ w * (maleundefined + femaleundefined)) /
(malemale + femalemale + malefemale + femalefemale +
+ w * (maleundefined + femaleundefined))
```

In Damegender, we have coded errors.py to implement the different definitions in diffrent apis.

The confusion matrix creates a matrix about the true and the guess. If you have this confusion matrix:

```
[[ 2, 0, 0]
 [ 0, 5, 0]]
```

It means, I have 2 females true and I've guessed 2 females and I've 5 males true and I've guessed 5 males. I don't have errors in my classifier.

```
[[ 2  1  0]
[ 2 14  0]]
```

It means, I have 2 females true and I've guessed 2 females and I've 14 males true and I've guessed 14 males. 1 female was considered male, 2 males was considered female.

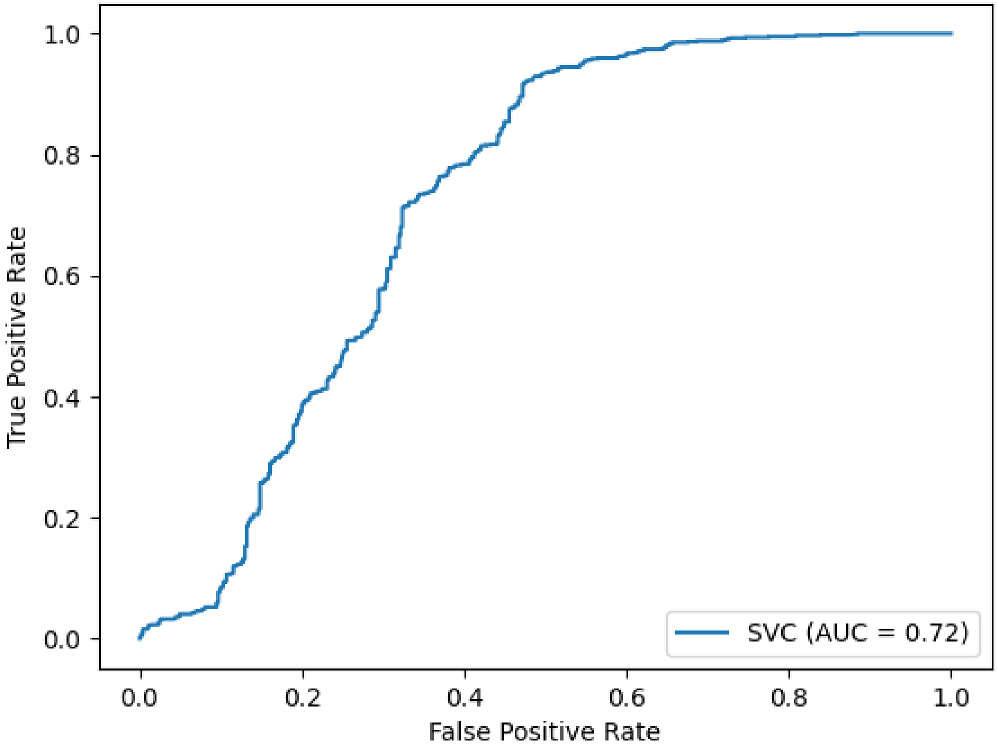
In Damegender, we have coded confusion.py to implement this concept with the different apis.

```
python3 confusion.py --csv=files/names/min.csv --api=damegender --jsondownloaded=files
A confusion matrix C is such that Ci,j is equal to the number of obser-
vations known to be in group i but predicted to be in group j.
If the classifier is nice, the diagonal is high because there are true positives
Damegender confusion matrix:

     M   F   U
M  [[ 5,  0,  0 ]
F   [ 0,  1,  0 ]]
```

Similar to confusion is ROC (Receiver Operating Characteristic) is a graphical plot that illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied. The ROC curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings.

In Damegender, you can use ROC relative to machine learning algorithms with the next command:

```
$ python3 roc.py svc
```



4.2 Principal Component Analysis (PCA)

4.2.1 Counting features in names

We have developed a script infofeatures.py with our datasets to visualize data about some features chosen by us.

```
$ python3 infofeatures.py ine
```

Take a look to the results with the different datasets:

Dataset	**Letter A**	**Last Letter A**	**Last Letter O**	**Last Letter Consonant**	**Last Letter Vocal**	**First Letter Consonant**	**First Letter Vocal**
Uruguay (females)	0.816	0.456	0.007	0.287	0.712	0.823	0.177
Uruguay (males)	0.643	0.249	0.062	0.766	0.234	0.771	0.228
Australia (females)	0.922	0.588	0.033	0.272	0.728	0.772	0.228

Australia (males)	0.818	0.03	0.269	0.57	0.43	0.763	0.237
Canada (females)	0.659	0.189	0.005	0.591	0.408	0.838	0.161
Canada (males)	0.752	0.22	0.025	0.54	0.456	0.818	0.181
Spain (females)	0.922	0.588	0.03	0.271	0.728	0.772	0.228
Spain (males)	0.818	0.03	0.268	0.569	0.43	0.763	0.236
United Kingdom (females)	0.825	0.374	0.013	0.322	0.674	0.765	0.235
United Kingdom (males)	0.716	0.036	0.039	0.78	0.218	0.799	0.2
USA (females)	0.816	0.456	0.007	0.287	0.712	0.823	0.177
USA (males)	0.643	0.02	0.061	0.765	0.234	0.84	0.159

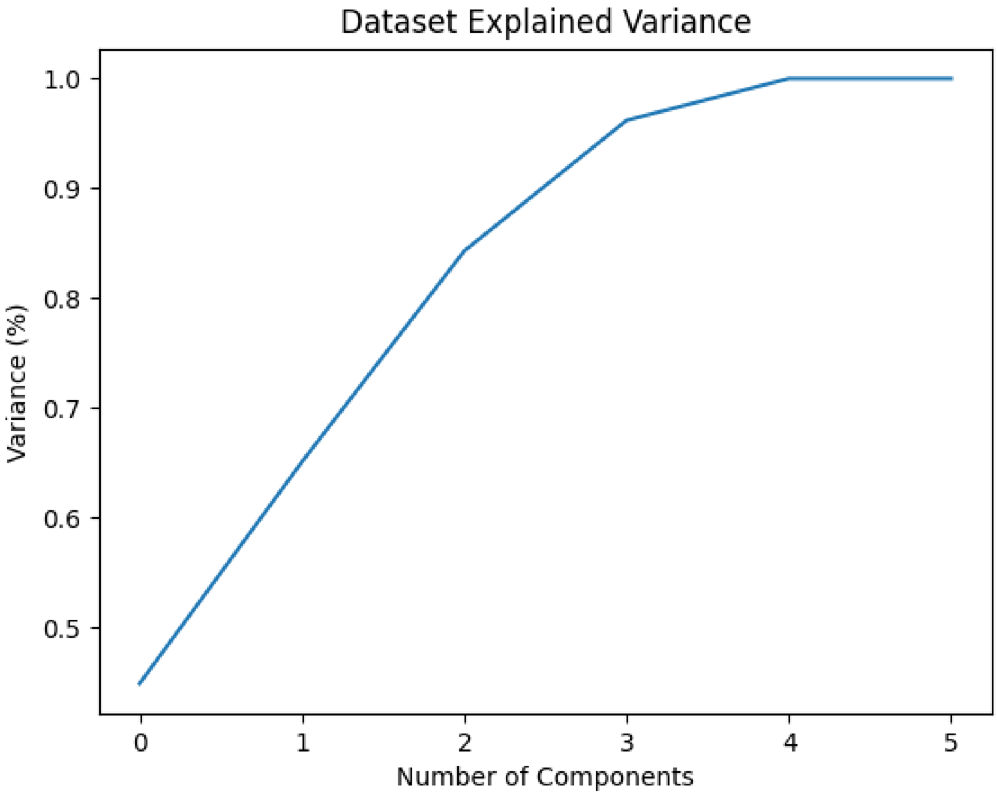
The countries where the main language is spanish (Uruguay + Spain) and english (USA + United Kingdom + Australia) are having very similar variation with the features chosen between males and females with these datasets (remember is the datasets extracted from official statistics provided by the states). Canada, a country french centric has different rules with this features.

The letter a is varying 0.2 from males to females in (USA and Uruguay) and 0.1 from males to females (United Kingdom, Australia and Spain). The last letter a is varying 0.5 from males to females in (Australia, Spain) around 0.4 in (USA, United Kingdom) and 0.2 in Uruguay. The last letter o from females to males is varying 0.2 in (Spain, Australia) and is equal in (Uruguay, USA, United Kingdom). For the last letter consonant all countries is giving the result that is for males, with results from 0.2 to 0.5: Uruguay and USA (0.5), United Kingdom (0.4), Australia and Spain (0.3). So last letter vocal is reverse tha last letter consonant. First letter consonant or first letter vocal is a non significative feature due to so similar results in english and spanish.

Surely, the rules it's a coincidence but we think that is a coincidence between languages due to that there are a good number of names to think different.

4.2.2 Choosing components

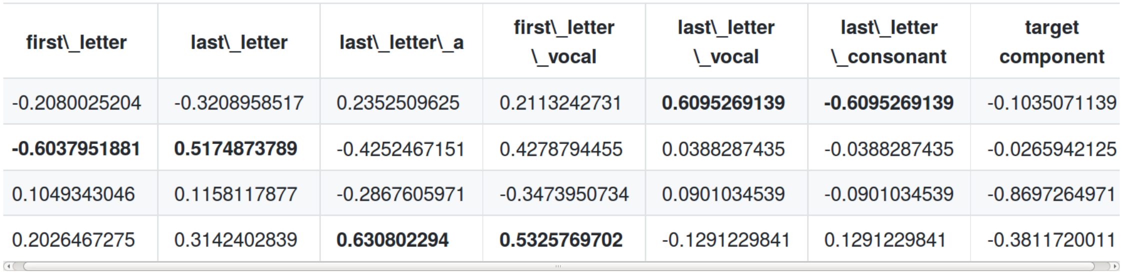
After, to choose features for our machine learning task, we can understand if this features makes sense with Principal Component Analysis. We have written 2 scripts for this task `pca-components.py` and `pca-features.py`. With `pca-components.py` we are giving a csv (files/features_list.csv, files/features_list_no_cat.csv, ...) and the output is an image where we can visualize a curve to determine when this curve stops the growth the number of components.



In the image, we can see that the curve stops the growth in the fourth component.

When you know the components you can execute `pca-features.py` so:

```
$ python3 pca-features.py  --categorical=both --components=4
The json file is created in files/pca.json
The html file is created in files/pca.html
```



first_letter	last_letter	last_letter_a	first_letter _vocal	last_letter _vocal	last_letter _consonant	target component
-0.2080025204	-0.3208958517	0.2352509625	0.2113242731	**0.6095269139**	**-0.6095269139**	-0.1035071139
-0.6037951881	**0.5174873789**	-0.4252467151	0.4278794455	0.0388287435	-0.0388287435	-0.0265942125
0.1049343046	0.1158117877	-0.2867605971	-0.3473950734	0.0901034539	-0.0901034539	-0.8697264971
0.2026467275	0.3142402839	**0.630802294**	**0.5325769702**	-0.1291229841	0.1291229841	-0.3811720011

To simplify and to learn, we can observe this analysis without letters. In this analysis, we can observe 4 components.

The first component is about if the last letter is vocal or consonant. If the last letter is vocal we can find a male and if the last letter is a consonant we can find a male.

The second component is about the first letter. The last letter is determining females and the first letter is determining males.

The third component is not giving relevant information.

The fourth component is giving the last_letter_a and the first_letter_vocal is for females.

5 Use Cases

5.1 Introduction

There are many research studies count males and females in specific communities such as Twitter, StackOverflow, ... We hope that with this manual software

A specific community has some clues to determine male or female, for example, in Twitter you observe the photo, nickname, real name, ...

5.2 Counting males and females in Debian

In the Debian community all member must have a gpg key to collaborate, so we can count males and females from the keyring. With gpg commands you can import a the debian keyring and dump the debian keyring in a csv file.

```
$ rsync -az --progress keyring.debian.org::keyrings/keyrings/ .
```

We have generated a script to count males and females:

```
~/git/damegender/src/damegender$ python3 count-debian-gender.py
Perhaps you need wait some minutes. You can take a tea or coffe now
debian males: 795
debian females: 24
```

In the dump of the debian keyring dataset we have divided name, surname and email in different fields. So, it's easy detect the name, although some names has several emails

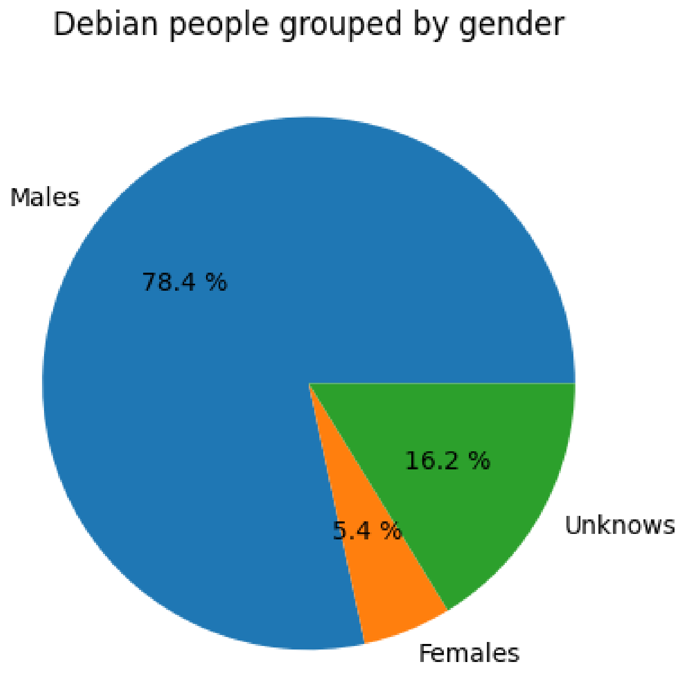
We have choosen the United States of America dataset and we are using the method name_frec to decide for male or female in the row. Take a look to the source:

```
import csv
import unicodedata
import unidecode
from pprint import pprint
import re
from app.dame_gender import Gender
from app.dame_utils import DameUtils

du = DameUtils()
g = Gender()

result=""
dm = []

with open('files/debian-maintainers-gpg-2020-04-01.csv') as csvfile:
    reader = csv.reader(csvfile, delimiter=',', quotechar='|')
    aux = ""
    cnt = 0
    for row in reader:
        cnt = cnt +1
        if (aux != row[0]):
            dm.append(row[0])
        aux = row[0]

print("Perhaps you need wait some minutes. You can take a tea or coffe now")

females = 0
males = 0
for rowdm in dm:
    if (int(g.name_frec(str(rowdm.upper()), ’us’)[’females’]) > int(g.name_frec(str(ro
        females = females + 1
    else:
        males = males + 1

print("debian males: %s" % males)
print("debian females: %s" % females)

csvfile.close()
```

The advantage using name_frec is about to understand how you are deciding male or female in the script counting males and females. In this script the decision is simple: a name is male if there are more males than females and female if there are more females than males.

The United States of America dataset is a good choice for Free Software communities, due to that this communities is based on english as main language and United States of

America is a leader country in software development. United States of America hosts people from different countries due to migrations towards good companies and universities.



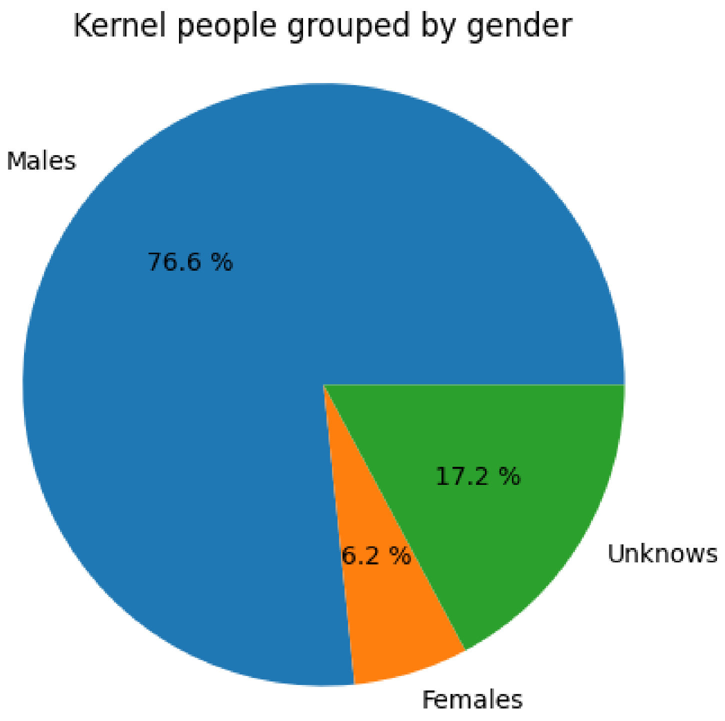
5.3 Counting males and females in Linux Kernel

When I'm writing this book the Linux Kernels maintainers appears in https://www.kernel.org/doc/html/latest/process/maintainers.html. Then I have downloaded the file and applied a single command:

```
cat maintainers.html | w3m -dump -T text/html | grep "Mail:" > maintainers.txt
```

You can makes fixes to this command from GNU/Emacs or with shell scripting. Later, you can apply:

```
$ python3 count-kernel.py
```



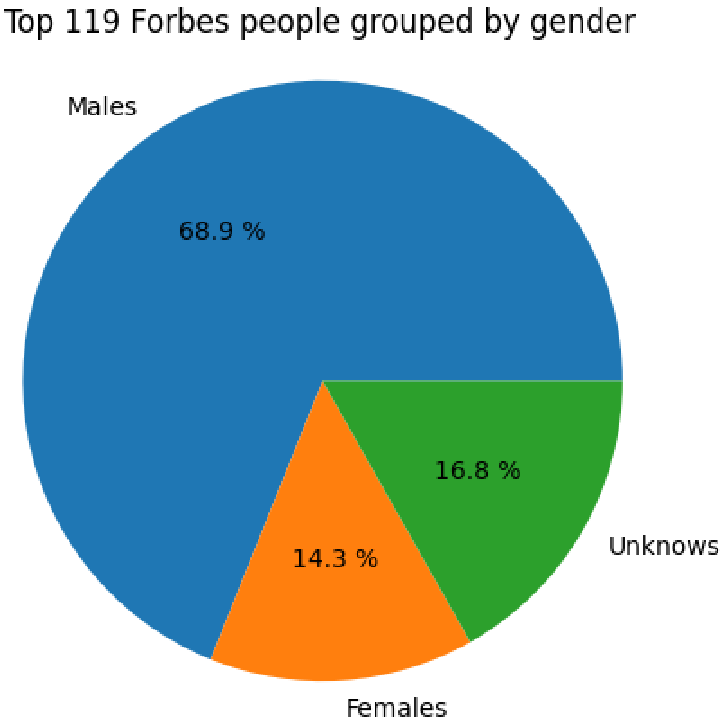
5.4 Counting males and females in Forbes

In the second example, we are using guess without machine learning instead of name_frec. If you are using guess you are trusting on damegender to take the decision, but perhaps you are not agree.

Please take a look about our guess method in the current state:

```
def guess(self, name, binary=False, *args, **kwargs):
    # guess list method
    dataset = kwargs.get('dataset', 'es')
    # guess method to check names dictionary
    guess = ''
    name = unidecode.unidecode(name).title()
    name.replace(name, "")
    dicc = self.name_frec(name, dataset)
    m = int(dicc['males'])
    f = int(dicc['females'])
    if ((m == 0) and (f == 0)):
        if binary:
            guess = 2
        else:
            guess = "unknown"
    elif (m > f):
        if binary:
            guess = 1
        else:
            guess = "male"
    elif (f > m):
        if binary:
            guess = 0
        else:
            guess = "female"
    else:
        if binary:
            guess = 2
        else:
            guess = "unknown"
    return guess
```

We are using the spanish dataset by default and the rest is the same idea that in the last script: more people using the name.



5.5 Deciding for males and females in images

There are many free software tools for decide gender in images files, we have selected the next tool:

```
$ git clone https://github.com/davidam/damephoto
$ cd damephoto/bin
$ python3 damephoto.py girl1.jpg
```

We can use this tool to decide gender about images from Twitter, Github, ...

5.6 Webscraping and Damegender because we want count scientifics

Sometimes, we can reach the database of names from a website, for example, we can retrieve a list of scientifics from Spain thanks to webometrics and the next script:

```
from lxml import html
import requests

print("Introduce an url from webometrics, for example, https://www.webometrics.info/en

import argparse

parser = argparse.ArgumentParser()
parser.add_argument("url", help="display the gender")
args = parser.parse_args()

page = requests.get(args.url)
tree = html.fromstring(page.content)

scientifics = tree.xpath('//tr/td/a/strong/text()')

print('Scientifics: %s' % scientifics)
```

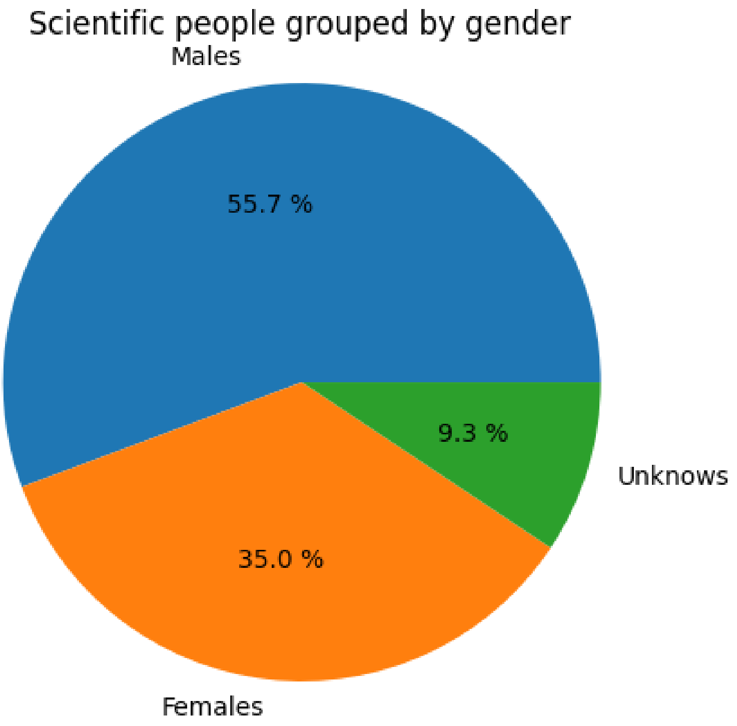
If you have retrieved the list of names in a file `files/scientifics.txt`, you could count males and females with the next script called `count-scientifics.py`:

```
import csv
import unicodedata
import unidecode
import re

from pprint import pprint
from app.dame_gender import Gender
from app.dame_utils import DameUtils
from ast import literal_eval
from app.dame_sexmachine import DameSexmachine

du = DameUtils()
g = Gender()
s = DameSexmachine()
```

```
with open('files/scientifics.txt') as f:
    mainlist = [list(literal_eval(line)) for line in f]

l = mainlist[0]

ll = []
for i in l:
    ll.append(i.split())


ten = ll[0:10]
hundred = ll[0:100]
thousand = ll[0:1000]

x = 0
y = 0
males = 0
females = 0
for j in hundred:
    if (len(j[0]) == 1):
        x = x + 1
    else:
        sex = g.guess(j[0], binary=False)
        y = y +1
        if (sex == "male"):
            males = males + 1
        elif (sex == "female"):
            females = females + 1

print("Number of scientifics with a single letter as first name: %s" % x)
print("Number of scientifics with the first name normal: %s" % y)
print("Number of females scientifics: %s" % females)
print("Number of males scientifics: %s" % males)

for j in thousand:
    if (len(j[0]) == 1):
        x = x + 1
    else:
        sex = g.guess(j[0], binary=False)
        y = y +1
        if (sex == "male"):
            males = males + 1
        else:
            females = females + 1

print("Number of females scientifics: %s" % females)
print("Number of males scientifics: %s" % males)
```

And the results are:

```
Number of females scientifics: 31425
Number of males scientifics: 47945
```



5.7 Counting males and females in a git repository

We can think a simple version of git2gender.py:

```
from app.dame_sexmachine import DameSexmachine
from app.dame_perceval import DamePerceval
from app.dame_utils import DameUtils
import sys
import argparse
parser = argparse.ArgumentParser()
parser.add_argument("url", help="Uniform Resource Link")
parser.add_argument('--directory')
parser.add_argument('--version', action='version', version='0.1')
args = parser.parse_args()
if (len(sys.argv) > 1):
    ds = DameSexmachine()
    du = DameUtils()
    dp = DamePerceval()
    l1 = dp.list_committers(args.url, args.directory)
    l2 = du.delete_duplicated(l1)
    l3 = du.clean_list(l2)

    females = 0
    males = 0
    unknowns = 0
    for g in l3:
        sm = ds.guess(g, binary=True)
        if (sm == 0):
            females = females + 1
        elif (sm == 1):
            males = males + 1
        else:
            unknowns = unknowns + 1

    print("The number of males sending commits is %s" % males)
    print("The number of females sending commits is %s" % females)
```

Try to execute this script:

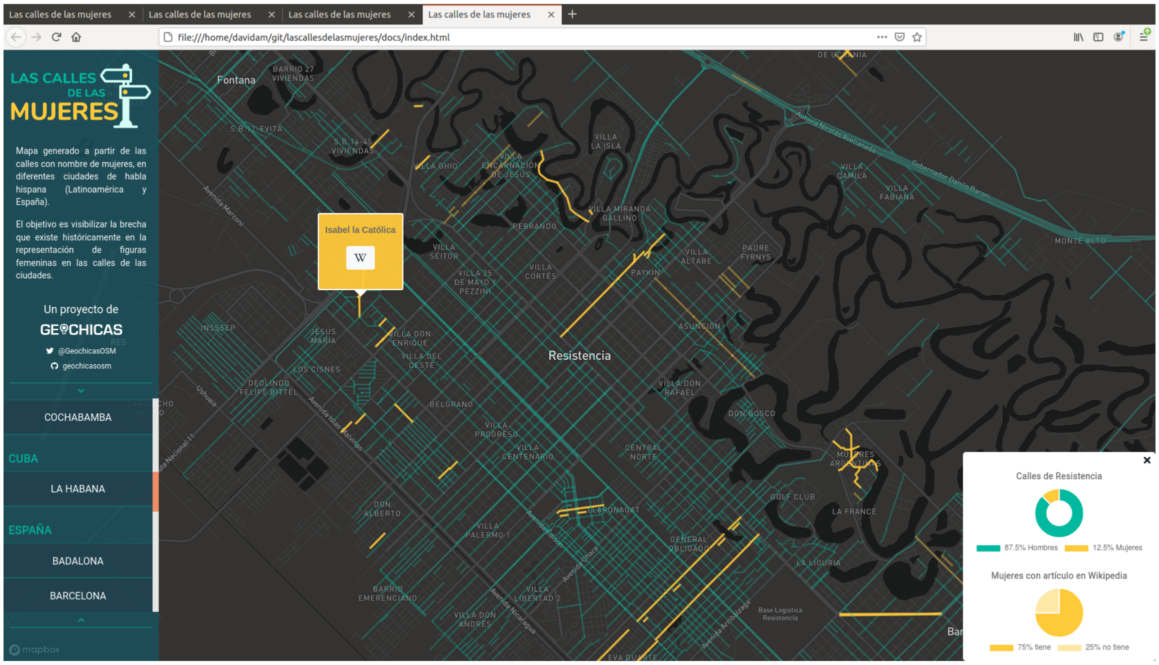
```
$ python3 git2gender.py https://github.com/davidam/davidam.git --directory="/tmp/clone
The number of males sending commits is 3
The number of females sending commits is 0
```

5.8 Counting males and females in Maps

Las calles de las mujeres is a project in NodeJS to display web streets with names about females using MapBox. You can download the project with:

```
$ git clone https://github.com/geochicasosm/lascallesdelasmujeres
```

It's licensed with a Creative Commons License (CC-BY-SA) `https://creativecommons.org/licenses/by`


Las calles de las mujeres
Las calles de las mujeres
Las calles de las mujeres
Las calles de las mujeres
file:///home/davidam/git/lascallesdelasmujeres/docs/index.html
LAS CALLES DE LAS MUJERES
Un proyecto de
GEOCHICAS
@GeochicasOSM
geochicasosm
COCHABAMBA
CUBA
LA HABANA
ESPAÑA
BADALONA
BARCELONA
mapbox
Isabel la Católica
W
Fontana
Resistencia
Calles de Resistencia
87.5% Hombres
12.5% Mujeres
Mujeres con artículo en Wikipedia
75% tiene
25% no tiene

Mapa generado a partir de las calles con nombre de mujeres, en diferentes ciudades de habla hispana (Latinoamérica y España).

El objetivo es visibilizar la brecha que existe históricamente en la representación de figuras femeninas en las calles de las ciudades.

It's giving statisticals about how many men and women has wikipedia article, too.

It has a strong community created by females (GeoChicas) `https://geochicas.github.io/` related with OpenStreetMap.

6 Theorical Frameworks

If you want do a social research for a quantitative study, such as, count males and females you can:

- Generate objectives about the research study
- To read and to understand previous works.
- Choose some theorical framework.
- Perhaps, with a qualitative study (interviews, focus groups, ...) you could understand better the problems, the specific vocabulary used by the people, the reasons about the decisions, ...

A theoretical framework consists of concepts and, together with their definitions and reference to relevant scholarly literature, existing theory that is used for your particular study. In the last chapters you have learnt to count males and females, but you need give meaning to the words that you are using about your gender study. That is the point in this chapter.

We present some theorical frameworks that you can use as example in your works:

- Philosophies about software market and freedoms
- Interculturalism and Multiculturalism
- Feminism, Ecofeminism and derivatives

6.1 Philosophies about software, market, freedom and gender

There are different philosophies developing software and we are counting males and females in Internet, so the floor is the software in this world. If we must analyze gender in a country the ideology is changing in the place where you are. In the software world is the same problem. So, we are giving the vocabulary and the philosophy for speak about software and ideologies.

The propietary software is the most common idea for the common people, operating systems such as Microsoft Windows or Mac OS. If you are using software with propietary licenses, the source files will be containing copyright notes such as:

```
# Copyright (C) 2020  David Arroyo Menéndez

# Author: David Arroyo Menéndez <davidam@gmail.com>
# Maintainer: David Arroyo Menéndez <davidam@gmail.com>

# All rights reserved
```

This idea is associated to big companies leading the market but any people can use this philosophy. The criticism appears with Richard Stallman about privacy and lack of freedom to the academic people, or hackers (people who knows read and write software and they do it for his objectives or global objectives). I could to say the monopoly is too strong with this license and the current social inertia and now nobody can change the market, we need another licenses to preserve the free market with an ethical strategy for startups and students.

Richard Stallman defines the Free Software with four freedoms: (0) to run the program, (1) to study and change the program in source code form, (2) to redistribute exact copies, and (3) to distribute modified versions.

This idea to build software as a social good and motivated by ethical values. The solution is to apply the GPL license and to request to GNU to include the software.

The copyright note in GNU would be similar to:

```
;; This software is free software: you can redistribute it and/or modify
;; it under the terms of the GNU General Public License as published by
;; the Free Software Foundation, either version 3 of the License, or
;; (at your option) any later version.

;; This software is distributed in the hope that it will be useful,
;; but WITHOUT ANY WARRANTY; without even the implied warranty of
;; MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
;; GNU General Public License for more details.

;; You should have received a copy of the GNU General Public License
;; along with GNU Emacs.  If not, see <https://www.gnu.org/licenses/>.
```

On opposition the Open Source movement believes in free licenses, but they thinks that the software is business and they want to develop Free Software by economy, so they prefer change the word Free Software by Open Source claiming their philosophy.

They redefines the Free Software Definition by the Open Source Definition:

```
1. Free Redistribution

The license shall not restrict any party from selling or giving away
the software as a component of an aggregate software distribution
containing programs from several different sources. The license shall
not require a royalty or other fee for such sale.

2. Source Code

The program must include source code, and must allow distribution in
source code as well as compiled form. Where some form of a product is
not distributed with source code, there must be a well-publicized
means of obtaining the source code for no more than a reasonable
reproduction cost, preferably downloading via the Internet without
charge. The source code must be the preferred form in which a
programmer would modify the program. Deliberately obfuscated source
code is not allowed. Intermediate forms such as the output of a
preprocessor or translator are not allowed.

3. Derived Works

The license must allow modifications and derived works, and must allow
them to be distributed under the same terms as the license of the
```

original software.

4. Integrity of The Author's Source Code

The license may restrict source-code from being distributed in modified form only if the license allows the distribution of "patch files" with the source code for the purpose of modifying the program at build time. The license must explicitly permit distribution of software built from modified source code. The license may require derived works to carry a different name or version number from the original software.

5. No Discrimination Against Persons or Groups

The license must not discriminate against any person or group of persons.

6. No Discrimination Against Fields of Endeavor

The license must not restrict anyone from making use of the program in a specific field of endeavor. For example, it may not restrict the program from being used in a business, or from being used for genetic research.

7. Distribution of License

The rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties.

8. License Must Not Be Specific to a Product

The rights attached to the program must not depend on the program's being part of a particular software distribution. If the program is extracted from that distribution and used or distributed within the terms of the program's license, all parties to whom the program is redistributed should have the same rights as those that are granted in conjunction with the original software distribution.

9. License Must Not Restrict Other Software

The license must not place restrictions on other software that is distributed along with the licensed software. For example, the license must not insist that all other programs distributed on the same medium must be open-source software.

10. License Must Be Technology-Neutral

```
No provision of the license may be predicated on any individual
technology or style of interface.
```

In the point six, we find the conflict with the feminist theories due to the possitive discrimination is a good idea to reach gender equity.

The GNU philosophy has the same problem explained on a different way. Only the Free Software is a good idea, if the software is not Free Software, then it's Propietary Software (the bad idea to avoid).

In Damegender, we want to deliver Free Software by practical reasons released with GPLv3 trough pypi.org and github.com the very popular sites to distribute Free Software written in Python. But we understand that we can to make positive discrimination in the development in favor to the women as an experiment with this copyright note in the development branch:

```
# You can share, copy and modify this software if you are a woman or you
# are David Arroyo Menéndez and you include this note.
```

This book will be published with this license, too.

Appendix A License

You can share, copy and modify this manual if you are a woman or you are David Arroyo Menéndez and you include this note.

The sources will be find in https://github.com/davidam/damegender/tree/master/manual

Index