# Damegender: Writing and Comparing Gender Detection Tools

David Arroyo Menéndez

January 27, 2020

# Abstract

Nowadays there are various APIs to detect gender from a name. In these slides, we offer a tool to use and compare these apis and a method to classify male and female applying machine learning and using a free license. The gender detection from a name is useful to make gender studies from social networks, mailing lists, software repositories, articles, etc.

# Download source and article to a make a good tracing

- git clone `https://github.com/davidam/damegender.git`
- pip3 install damegender[all]

# Social Need

In this moment there are a gender gap between males and females in computer science and science in general (STEMM: Science, Technology, Engineering, Mathematics and Medicine). Create free tools and improve the current state of art allows measure and later create policies with facts to fix the situation.

# Problems addressed

- Evaluate quality/price with different commercial solutions.
- Think about free solutions.
- Treatment with names without census (ex: nicknames, diminutives, ...)
- Massive gender detection in Internet (ex: mailing lists, software repositories, ...)

# Damegender as Python solution

- Evaluate quality of different solutions applying metrics suggested by Santamaría and Mihaljević

(2018)

- We have developed a tool to give support to Spanish and English from the census to make the gender

detection to understand the problems.

- We have developed a machine learning solution to strings not found in the census dataset.
- We have proof-of-concept tests applying Perceval to detect gender in mailing lists and software

repositories.

# State of Art

- Comparing APIs
- Datasets
- Free Software
- More about gender (image recognition, hand written, speeches, . . . )
- Massive gender detection

# APIs (Market)

| | Gender API | gender-guesser | genderize.io | NameAPI | NamSor | damegender |
|---|---|---|---|---|---|---|
| Database size | 431322102 | 45376 | 114541298 | 1428345 | 4407502834 | 57282 |
| Regular data updates | yes | no | no | yes | yes | yes, developing |
| Handles unstructured full name strings | yes | no | no | yes | no | yes |
| Handles surnames | yes | no | no | yes | yes | yes |
| Handles non-Latin alphabets | partially | no | partially | yes | yes | no |
| Implicit geo-localization | yes | no | no | yes | yes | no |
| Exists locale | yes | yes | yes | yes | yes | yes |
| Assingment type | probilistic | binary | probabilistic | probabilistic | probabilistic | probabilistic |
| Free parameters | $\text{total}_{\text{names}}$, probability | gender | probability, count | confidence | scale | $\text{total}_{\text{names}}$, count |
| Prediction | no | no | no | no | no | yes |
| Free license | no | yes | no | no | no | yes |
| API | yes | no | yes | yes | yes | future |
| free requests limited | yes (200) | unlimited | yes | yes | yes | unlimited |

# Measuring success in gender detection tools from the name

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)
Accuracy 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))

# Damegender to measure precision, recall, accuracy or f1 score

In Damegender, we are using accuracy.py with the different measures (precision, recall, accuracy and f1 score) in different apis from an input. For instance:

```
$ python3 accuracy.py --api="damegender" --measure="recall"
--csv=files/names/allnoundefined.csv
$ python3 accuracy.py --api="nameapi" --measure="precision"
--csv=files/names/allnoundefined.csv
--jsondownloaded=files/names/nameapifiles_names_allnoundefined.
$ python3 accuracy.py --api="namsor" --measure="accuracy"
--csv=files/names/allnoundefined.csv
--jsondownloaded=files/names/namsorfiles_names_allnoundefined.c
```

# Accuracy results with Damegender

| Name | Accuracy | Precision | F1score | Recall |
|---|---|---|---|---|
| Genderapi | 0.969 | 0.972 | 0.964 | 1.0 |
| Genderize | 0.927 | 0.976 | 0.965 | 1.0 |
| Namsor | 0.867 | 0.973 | 0.924 | 1.0 |
| Nameapi | 0.83 | 0.974 | 0.9 | 1.0 |
| Gender Guesser | 0.774 | 0.985 | 0.872 | 1.0 |
| Damegender | 0.745 | 0.879 | 0.879 | 1.0 |

# Measuring errors in gender detection tools from the name

Error coded defines if the true is different than the guessed. That's divide the number of elements with errors against the total number of elements:

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

Error coded without na defines if the true is different than the guessed, but without undefined results. That's divide the number of elements with undefined errors against the total number of elements

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

Error gender bias allows to understand if the error is bigger guessing males than females or vice versa. That's males guessed as females minus females guessed as males and this number divided divided by the total number of elements not guessed as undefined.

# Damegender to measure errors

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

```
$ python3 errors.py --api="damegender" --csv=files/names/allnou
Damegender with files/names/allnoundefined.csv has:
+ The error code: 0.2547594323295258
+ The error code without na: 0.2547594323295258
+ The na coded: 0.0
+ The error gender bias: -0.04949809622706819
```

# Errors results with Damegender

| API | err code | err without na | na coded | err gender bias |
|---|---|---|---|---|
| Damegender | 0.255 | 0.255 | 0.0 | -0.049 |
| GenderApi | 0.167 | 0.167 | 0.0 | -0.167 |
| Gender Guesser | 0.225 | 0.027 | 0.204 | 0.003 |
| Namsor | 0.167 | 0.167 | 0.0 | 0.167 |
| Nameapi | 0.361 | 0.267 | 0.129 | 0.001 |

# Confusion Matrix to measure success and errors

The rows of the data source element are true and in the columns the elements are identified as guess.

```
[[ 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.

# Damegender and Confusion Matrix

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

```
$ python3 downloadjson.py --api="namsor"
--csv=files/names/allnoundefined.csv
$ python3 confusion.py --api="namsor"
--csv=files/names/allnoundefined.csv
--jsondownloaded=files/names/namsorfiles_names_allnoundefined.c

A confusion matrix C is such that Ci,j is equal to the number o
If the classifier is nice, the diagonal is high because there a
Namsor confusion matrix:

[[ 3325, 139, 346 ]
 [ 78, 1686, 204 ]]
```

# Confusion Matrix results with Damegender

| APIs | | m | f | u |
|---|---|---|---|---|
| Genderapi | m | 3589 | 155 | 67 |
| | f | 211 | 1734 | 23 |
| Genderguesser | m | 3326 | 139 | 346 |
| | f | 78 | 1686 | 204 |
| Genderize | m | 3157 | 242 | 412 |
| | f | 75 | 1742 | 151 |
| Nameapi | m | 2627 | 674 | 507 |
| | f | 667 | 1061 | 240 |
| Namsor | m | 3325 | 139 | 346 |
| | f | 78 | 1686 | 204 |
| Damegender | m | 3033 | 778 | 0 |
| | f | 276 | 1692 | 0 |

# Datasets

## Open Data

- Wikidata
- Spain. INE (Instituto Nacional de Estadística)
- United Kingdom Census
- United States of America Census
- NLTK

## APIs

- NameApi
- GenderApi
- Genderize
- Namsor

# Implementation (I). Scientific requirements

- Scikit
- NLTK
- Numpy
- Matplotlib
- Perceval

# Implementation (II). Features

- To know the gender about a name in Spanish or English (current status) from open census in local.
- Decide about males and females in strings using different machine learning algorithms.
- To use the main solutions in gender detection (genderize, genderapi, namsor, nameapi and gender guesser) from a command and downloading results in a json file
- To research with statistics about why a name is related with males or females. We provide python commands about study and compare gender solutions with: confusion matrix, accuracies, error measures. And decide about features: statistical feature weight, pca about features, ...
- Determine gender gap in free software repositories or mailing lists

(proof of concept)

# Conclusions

- Market = APIs
- Future Market = Free Software and Open Data?
- Machine Learning can help to classify nicknames or diminutives
- About massive gender detection

# License