MLConjug Documentation

Versiune 3.4.0

SekouD

apr. 29, 2019

Cuprins

Cuprins:

CAPITOLUL 1

mlconjug

O bibliotecă Python care conjugă verbe în franceză, engleză, spaniolă, italiană, portugheză și română (mai curând) folosind tehnici de învățare a mașinilor”
Orice verb într-unul din limbile acceptate poate fi conjugat, deoarece modulul conține un model de învățare a mașinilor despre cum se comportă verbul”
Chiar și verbele complet noi sau confecționate pot fi conjugate cu succes în acest fel”
Modelele pre-instruite furnizate sunt compuse din:

- un extractor de caracteristici binare
- un selector de elemente care utilizează clasificarea vectorială de sprijin liniar,
- un clasificator care utilizează coborârea stochastică.

MLConjug utilizează scikit-learn pentru a implementa algoritmii Machine Learning.
Users of the library can use any compatible classifiers from scikit-learn to modify and retrain the models.

The training data for the french model is based on Verbiste https://perso.b2b2c.ca/~sarrazip/dev/verbiste.html .
Datele de antrenament pentru limbile engleză, spaniolă, italiană, portugheză și română au fost generate folosind tehnici de învățare nesupravegheate, utilizând modelul francez, ca model de interogare în timpul antrenamentului.

- Software-ul gratuit: licență MIT
- Documentație: https://mlconjug.readthedocs.io.

1.1 Limbi acceptate

- Limba franceza
- Engleză
- Spaniolă
- Italiană
- Portugheză
- Română

1.2 Caracteristici

- PI ușor de utilizat
- Include modele pre-instruite cu o acuratețe de 99% + în prezicerea clasei de conjugare a verbelor necunoscute.
- Formați cu ușurință modele noi sau adăugați limbi noi.
- Integrați cu ușurință programul MLConjug în propriile proiecte.
- Poate fi folosit ca un instrument de linie de comandă.

1.3 Credite

Acest pachet a fost creat cu ajutorul Verbiste și scikit-learn"

Sigla a fost creată de Zuur.

CAPITOLUL 2

Instalare

2.1 Versiune stabila

Pentru a instala MLConjug, executați această comandă în terminalul dvs.:

```
$ pip install mlconjug
```

Aceasta este metoda preferată de a instala MLConjug, deoarece va instala întotdeauna cea mai recentă versiune stabilă.

Dacă nu aveți instalat" pip „, acest" ghid de instalare Python „vă poate ghida prin acest proces.

2.2 Din surse

Sursele pentru MLConjug pot fi descărcate din" Github repo „.

Puteți fie să clonați depozitul public:

```
$ git clone git://github.com/SekouD/mlconjug
```

Sau descarca" tarball „_:

```
$ curl  -OL https://github.com/SekouD/mlconjug/tarball/master
```

Odată ce aveți o copie a sursei, o puteți instala cu:

```
$ python setup.py install
```

CAPITOLUL 3

Utilizare

Notă: Limba implicită este limba franceză. Atunci când este chemată fără a specifica o limbă, biblioteca va încerca să conjugă verbul în franceză”

Pentru a folosi MLConjug într-un proiect cu modelele de conjugare pre-instruite furnizate

```
import mlconjug

# To use mlconjug with the default parameters and a pre-trained conjugation model.
default_conjugator = mlconjug.Conjugator(language='fr')

# Verify that the model works
test1 = default_conjugator.conjugate("manger").conjug_info['Indicatif']['Passé Simple
↪']['1p']
test2 = default_conjugator.conjugate("partir").conjug_info['Indicatif']['Passé Simple
↪']['1p']
test3 = default_conjugator.conjugate("facebooker").conjug_info['Indicatif']['Passé␣
↪Simple']['1p']
test4 = default_conjugator.conjugate("astigratir").conjug_info['Indicatif']['Passé␣
↪Simple']['1p']
test5 = default_conjugator.conjugate("mythoner").conjug_info['Indicatif']['Passé␣
↪Simple']['1p']
print(test1)
print(test2)
print(test3)
print(test4)
print(test5)

# You can now iterate over all conjugated forms of a verb by using the newly added␣
↪Verb.iterate() method.
default_conjugator = mlconjug.Conjugator(language='en')
test_verb = default_conjugator.conjugate("be")
all_conjugated_forms = test_verb.iterate()
print(all_conjugated_forms)
```

Pentru a folosi MLConjug într-un proiect și a antrena un nou model

```
# Set a language to train the Conjugator on
lang = 'fr'

# Set a ngram range sliding window for the vectorizer
ngrange = (2,7)

# Transforms dataset with CountVectorizer. We pass the function extract_verb_features␣
↪to the CountVectorizer.
vectorizer = mlconjug.CountVectorizer(analyzer=partial(mlconjug.extract_verb_features,
↪ lang=lang, ngram_range=ngrange),
                              binary=True)

# Feature reduction
feature_reductor = mlconjug.SelectFromModel(mlconjug.LinearSVC(penalty="l1", max_
↪iter=12000, dual=False, verbose=0))

# Prediction Classifier
classifier = mlconjug.SGDClassifier(loss="log", penalty='elasticnet', l1_ratio=0.15,␣
↪max_iter=4000, alpha=1e-5, random_state=42, verbose=0)

# Initialize Data Set
dataset = mlconjug.DataSet(mlconjug.Verbiste(language=lang).verbs)
dataset.construct_dict_conjug()
dataset.split_data(proportion=0.9)

# Initialize Conjugator
model = mlconjug.Model(vectorizer, feature_reductor, classifier)
conjugator = mlconjug.Conjugator(lang, model)

#Training and prediction
conjugator.model.train(dataset.train_input, dataset.train_labels)
predicted = conjugator.model.predict(dataset.test_input)

# Assess the performance of the model's predictions
score = len([a == b for a, b in zip(predicted, dataset.test_labels) if a == b]) /␣
↪len(predicted)
print('The score of the model is {0}'.format(score))

# Verify that the model works
test1 = conjugator.conjugate("manger").conjug_info['Indicatif']['Passé Simple']['1p']
test2 = conjugator.conjugate("partir").conjug_info['Indicatif']['Passé Simple']['1p']
test3 = conjugator.conjugate("facebooker").conjug_info['Indicatif']['Passé Simple'][
↪'1p']
test4 = conjugator.conjugate("astigratir").conjug_info['Indicatif']['Passé Simple'][
↪'1p']
test5 = conjugator.conjugate("mythoner").conjug_info['Indicatif']['Passé Simple']['1p
↪']
print(test1)
print(test2)
print(test3)
print(test4)
print(test5)

# Save trained model
with open('path/to/save/data/trained_model-fr.pickle', 'wb') as file:
    pickle.dump(conjugator.model, file)
```

Pentru a folosi MLConjug din linia de comandă

```
$ mlconjug manger

$ mlconjug bring -l en

$ mlconjug gallofar --language es
```

CAPITOLUL 4

Pachetul Api Documentația pentru mlconjug

4.1 Referință API pentru clasele din mlconjug.mlconjug.py

Modul principal MLConjug.

Acest modul declară clasele principale cu care interacționează utilizatorul.

Modulul definește clasele necesare pentru a interconecta cu modelele Machine Learning.

mlconjug.mlconjug.**extract_verb_features**(*verb*, *lang*, *ngram_range*)

Vectorizatorul personalizat optimizat pentru extragerea caracteristicilor verbelor.
Subclasele Vectorizer sklearn.feature_extraction.text.CountVectorizer.
As in Indo-European languages verbs are inflected by adding a morphological suffix, the vectorizer extracts verb endings and produces a vector representation of the verb with binary features.

Pentru a îmbunătăți rezultatele extragerii de elemente, au fost incluse și alte caracteristici:

Caracteristicile sunt n-gramele de nobile ale notelor, începând n-grame, lungimea verbului, numărul de vocale, numărul de consoane și raportul dintre vocale pe consoane.

Parametrii

- **verb** – șir. Verb să vectorizeze.
- **lang** – string. Language to analyze.

- **ngram_range** – tuplu. Intervalul ferestrei glisante ngram.

Întoarce listă. Lista celor mai importante trăsături ale verbului pentru sarcina de a găsi clasa lui de conjugare.

class mlconjug.mlconjug.**Conjugator**(*language='fr'*, *model=None*)

Aceasta este clasa principală a proiectului.
The class manages the Verbiste data set and provides an interface with the scikit-learn pipeline.
If no parameters are provided, the default language is set to french and the pre-trained french conjugation pipeline is used.
Clasa definește metoda conjugată (verb, limbă) care este principala metodă a modulului.

Parametrii

- **language** – Limbajul conjugatorului. Limba implicită este" fr „pentru franceză.
- **model** – mlconjug.Model sau scikit-learn Pipeline sau Classifier implementând metodele fit () și predictive (). O conductă furnizată de utilizator în cazul în care utilizatorul și-a pregătit propria conductă.

conjugate(*verb*, *subject='abbrev'*)

Aceasta este principala metodă a acestei clase.
Mai întâi verifică dacă verbul este în Verbiste"
If it is not, and a pre-trained scikit-learn pipeline has been supplied, the method then calls the pipeline to predict the conjugation class of the provided verb.

Returnează un obiect Verb sau Nici unul.

Parametrii

- **verb** – șir. Verbul să conjugați.
- **subject** – string" („string"), controlează denumirile abreviate sau complet, valoarea implicită este „abbrev"

Întoarce Obiect Verb sau Niciunul.

set_model(*model*)
Assigns the provided pre-trained scikit-learn pipeline to be able to conjugate unknown verbs.

Parametrii **model** – clasificator sau conducte scikit-learn"

class mlconjug.mlconjug.**DataSet**(*verbs_dict*)

Această clasă deține și gestionează setul de date.
Defines helper methodss for managing Machine Learning tasks like constructing a training and testing set.

Parametrii **verbs_dict** – Un dicționar de verbe și clasa lor de conjugare corespunzătoare.

construct_dict_conjug()

Populează dicționarul care conține șabloanele de conjugare.

Populează listele care conțin verbele și șabloanele lor”

split_data (*threshold=8*, *proportion=0.5*)

Împărțiți datele într-un antrenament și un set de testare.

Parametrii

- **threshold** – int Dimensiunea minimă a clasei de conjugare care trebuie divizată.
- **proportion** – float” Proporția probelor din setul de antrenament trebuie să fie între 0 și 1.

class mlconjug.mlconjug.**Model** (*vectorizer=None*, *feature_selector=None*, *classifier=None*, *language=None*)

Bazele:: clasa:‘ obiect‘

This class manages the scikit-learn pipeline.

Conducta include un vectorizator de caracteristici, un selector de caracteristici și un clasificator.

If any of the vectorizer, feature selector or classifier is not supplied at instance declaration, the __init__ method will provide good default values that get more than 92% prediction accuracy.

Parametrii

- **vectorizer** – scikit-learn Vectorizer”
- **feature_selector** – clasificator scikit-learn cu o metodă fit_transform ()
- **classifier** – clasificator Scikit-learn cu o metodă predictivă ()
- **language** – limbajul corpusului de verbe care urmează să fie analizat.

train (*samples*, *labels*)

Trains the pipeline on the supplied samples and labels.

Parametrii

- **samples** – ista de verbe.
- **labels** – Listă de șabloane de verb.

predict (*verbs*)

Prezice clasa de conjugare a listei furnizate de verbe”

Parametrii **verbs** – ista de verbe.

Întoarce ista de grupuri prezise de conjugare.

4.2 Referință API pentru clasele din mlconjug.PyVerbiste.py

PyVerbiste.

O bibliotecă Python pentru conjugarea verbelor în franceză, engleză, spaniolă, italiană, portugheză și română (mai curând)”

Acesta conține date de conjugare generate de modelele de învățare mecanică folosind librăria python mlconjug.

Mai multe informații despre mlconjug la adresa https://pypi.org/project/mlconjug/

Datele de conjugare sunt conforme cu schema XML definită de Verbiste.
Mai multe informații despre Verbiste la adresa https://perso.b2b2c.ca/~sarrazip/dev/conjug_manager.html

class mlconjug.PyVerbiste.**ConjugManager**(*language='default'*)
This is the class handling the mlconjug json files.

Parametrii **language** – string. | The language of the conjugator. The default value is fr for French. | The allowed values are: fr, en, es, it, pt, ro.

_load_verbs(*verbs_file*)
Load and parses the verbs from the json file.

Parametrii **verbs_file** – string or path object. Path to the verbs json file.

_load_conjugations(*conjugations_file*)
Load and parses the conjugations from the xml file.

Parametrii **conjugations_file** – șir sau obiect cale. Calea spre fișierul xml de conjugare.

_detect_allowed_endings()

Detectează terminațiile permise pentru verbe în limbile acceptate.
Toate limbile acceptate, cu excepția limbii engleze, restricționează forma pe care o poate lua un verb"
Deoarece engleza este mult mai productivă și mai variată în morfologia verbelor ei, orice cuvânt este permis ca verb"

Întoarce un set care conține terminațiile permise ale verbelor în limba țintă.

is_valid_verb(*verb*)

Verifică dacă verbul este un verb valabil în limba dată.
Cuvintele în limba engleză sunt tratate întotdeauna ca verbe posibile.
Verbe în alte limbi sunt filtrate după terminările lor.

Parametrii **verb** – Verbul de a conjuga.

Întoarce Adevărat dacă verbul este un verb valabil în limba respectivă

get_verb_info(*verb*)
Obține informații verbale și returnează o instanță VerbInfo.

Parametrii **verb** – șir. Verbul să conjugați.

Întoarce Obiect VerbInfo sau Nici unul.

get_conjug_info(*template*)
Obține informații de conjugare corespunzătoare șablonului dat.

Parametrii **template** – șir. Numele verbului care se termină.

Întoarce OrdonatDict sau Nici unul. OrdonatDict care conține sufixele conjugate ale șablonului.

class mlconjug.PyVerbiste.**Verbiste**(*language='default'*)
Bases: *mlconjug.PyVerbiste.ConjugManager*

Aceasta este clasa care gestionează fișierele xml Verbiste.

Parametrii **language** – string. | The language of the conjugator. The default value is fr for French. | The allowed values are: fr, en, es, it, pt, ro.

_load_verbs(*verbs_file*)
Load and parses the verbs from the xml file.

Parametrii **verbs_file** – șir sau obiect cale. Calea spre fișierul xml verbale.

_parse_verbs(*file*)
Parses the XML file.

Parametrii **file** – Fișier XML conținând verbele"

Întoarce OrderedDict Un ordin ordonat care conține verbul și șablonul său pentru toate verbele din fișier.

_load_conjugations(*conjugations_file*)
Load and parses the conjugations from the xml file.

Parametrii **conjugations_file** – șir sau obiect cale. Calea spre fișierul xml de conjugare.

_parse_conjugations(*file*)
Parses the XML file.

Parametrii **file** – FileObject, fișier XML care conține șabloanele de conjugare.

Întoarce OrderedDict: Un ordin ordonat care conține toate șabloanele de conjugare din fișier.

_load_tense(*tense*)
Încarcă și analizează formele inflexionate ale timpului din fișierul xml.

Parametrii **tense** – listă de etichete xml care conțin formulare inflexionate. Lista formelor inflexionate pentru procesarea momentului curent.

Întoarce list. List of inflected forms.

_detect_allowed_endings()

Detectează terminațiile permise pentru verbe în limbile acceptate.
Toate limbile acceptate, cu excepția limbii engleze, restricționează forma pe care o poate lua un verb"
Deoarece engleza este mult mai productivă și mai variată în morfologia verbelor ei, orice cuvânt este permis ca verb"

Întoarce un set care conține terminațiile permise ale verbelor în limba țintă.

get_conjug_info(*template*)
Obține informații de conjugare corespunzătoare șablonului dat.

Parametrii **template** – șir. Numele verbului care se termină.

Întoarce OrdonatDict sau Nici unul. OrdonatDict care conține sufixele conjugate ale șablonului.

get_verb_info(*verb*)
Obține informații verbale și returnează o instanță VerbInfo.

Parametrii **verb** – șir. Verbul să conjugați.

Întoarce Obiect VerbInfo sau Nici unul.

is_valid_verb(*verb*)

Verifică dacă verbul este un verb valabil în limba dată.
Cuvintele în limba engleză sunt tratate întotdeauna ca verbe posibile.
Verbe în alte limbi sunt filtrate după terminările lor.

Parametrii **verb** – Verbul de a conjuga.

Întoarce Adevărat dacă verbul este un verb valabil în limba respectivă

class mlconjug.PyVerbiste.**VerbInfo**(*infinitive*, *root*, *template*)
Această clasă definește structura de informații conjug_manager Verbiste.

Parametrii

- **infinitive** – șir. Formă infinitivă a verbului.
- **root** – rădăcină. Rădăcina lexicală a verbului"
- **template** – șir. Numele verbului care se termină.

class mlconjug.PyVerbiste.**Verb**(*verb_info*, *conjug_info*, *subject='abbrev'*, *predicted=False*)
This class defines the Verb Object. TODO: Make the conjugated forms iterable by implementing the iterator protocol.

Parametrii

- **verb_info** – Obiectul VerbInfo.
- **conjug_info** – OrderedDict.
- **subject** – string" („string"), controlează denumirile abreviate sau complet, valoarea implicită este „abbrev"
- **predicted** – bool. Indică dacă informațiile despre conjugare au fost prezise de model sau au fost extrase din setul de date.

iterate()
Iterates over all conjugated forms and returns a list of tuples of those conjugated forms. :return:

_load_conjug()

Populează formele inflaționate ale verbului"
Aceasta este versiunea generică a acestei metode.
Nu adaugă pronume personale forțelor conjugate"
Această metodă se poate ocupa de orice limbă nouă, dacă structura de conjugare este conformă cu Schema XML Verbiste"

class mlconjug.PyVerbiste.**VerbFr**(*verb_info*, *conjug_info*, *subject='abbrev'*, *predicted=False*)
Bazele:: clasa:" mlconjug.PyVerbiste.Verb'

Această clasă definește obiectul verbal francez.

_load_conjug()

Populează formele inflaționate ale verbului"
Adăugă pronume personale la verbele inflaționate.

iterate()
Iterates over all conjugated forms and returns a list of tuples of those conjugated forms. :return:

class mlconjug.PyVerbiste.**VerbEn**(*verb_info*, *conjug_info*, *subject='abbrev'*, *predicted=False*)
Bazele:: clasa:" mlconjug.PyVerbiste.Verb'

Această clasă definește Obiectul englez de verb.

_load_conjug()

Populează formele inflaționate ale verbului”
Adăugă pronume personale la verbele inflaționate.

iterate()
Iterates over all conjugated forms and returns a list of tuples of those conjugated forms. :return:

class mlconjug.PyVerbiste.**VerbEs**(*verb_info*, *conjug_info*, *subject='abbrev'*, *predicted=False*)
Bazele:: clasa:” mlconjug.PyVerbiste.Verb‘

Această clasă definește Obiectul verbului spaniol.

_load_conjug()

Populează formele inflaționate ale verbului”
Adăugă pronume personale la verbele inflaționate.

iterate()
Iterates over all conjugated forms and returns a list of tuples of those conjugated forms. :return:

class mlconjug.PyVerbiste.**VerbIt**(*verb_info*, *conjug_info*, *subject='abbrev'*, *predicted=False*)
Bazele:: clasa:” mlconjug.PyVerbiste.Verb‘

Această clasă definește obiectul verbului italian”

_load_conjug()

Populează formele inflaționate ale verbului”
Adăugă pronume personale la verbele inflaționate.

iterate()
Iterates over all conjugated forms and returns a list of tuples of those conjugated forms. :return:

class mlconjug.PyVerbiste.**VerbPt**(*verb_info*, *conjug_info*, *subject='abbrev'*, *predicted=False*)
Bazele:: clasa:” mlconjug.PyVerbiste.Verb‘

Această clasă definește obiectul verbului portughez.

_load_conjug()

Populează formele inflaționate ale verbului”
Adăugă pronume personale la verbele inflaționate.

iterate()
Iterates over all conjugated forms and returns a list of tuples of those conjugated forms. :return:

class mlconjug.PyVerbiste.**VerbRo**(*verb_info*, *conjug_info*, *subject='abbrev'*, *predicted=False*)
Bazele:: clasa:” mlconjug.PyVerbiste.Verb‘

Această clasă definește obiectul verbului românesc”

iterate()
Iterates over all conjugated forms and returns a list of tuples of those conjugated forms. :return:

_load_conjug()

Populează formele inflaționate ale verbului”
Adăugă pronume personale la verbele inflaționate.

CAPITOLUL 5

Contribuind

Contribuțiile sunt binevenite și sunt foarte apreciate! Fiecare ajutor mic și creditul vor fi întotdeauna acordate”

Puteți contribui în multe feluri:

5.1 Tipuri de contribuții

5.1.1 Raportează bug-uri

Raportați bug-uri la adresa https://github.com/SekouD/mlconjug/issues”

Dacă raportați un bug, vă rugăm să includeți:

- Numele și versiunea sistemului de operare.
- Orice detalii despre configurația dvs. locală care ar putea fi de ajutor în rezolvarea problemelor.
- Pași detaliați pentru a reproduce bug-ul.

5.1.2 Fix Bugs

Uita-te prin problemele GitHub pentru bug-uri. Orice etichetat cu ” bug „si ” help wanted „este deschis oricui vrea sa o implementeze.

5.1.3 Caracteristicile implementării

Uitați-vă la problemele GitHub pentru caracteristici. Orice etichetat cu ” îmbunătățire „și ” help wanted „este deschis oricui dorește să îl implementeze.

5.1.4 Scrierea documentației

MLConjug ar putea folosi întotdeauna mai multă documentație, fie ca parte a documentelor oficiale MLConjug, în docstrings, sau chiar pe web în bloguri, articole și altele.

5.1.5 Trimite parerea ta

Cea mai bună modalitate de a trimite feedback este să trimiteți o problemă la adresa https://github.com/SekouD/mlconjug/issues.

Dacă propui o caracteristică:

- Explicați în detaliu modul în care ar funcționa.
- Țineți cât mai restrâns domeniul de aplicare, pentru a facilita implementarea acestuia"
- Amintiți-vă că acesta este un proiect bazat pe voluntari și că contribuțiile sunt binevenite :)

5.2 Incepe!

Ați venit să contribuiți? Iată cum puteți crea" mlconjug „pentru dezvoltarea locală.

1. Reportați repo-ul" mlconjug „pe GitHub.
2. Clonează-ți furca pe plan local

```
$ git clone git@github.com:your_name_here/mlconjug.git
```

3. Instalați-vă copia locală într-o virtualenv. Presupunând că aveți instalat virtualenvwrapper, acesta este modul în care vă configurați furculița pentru dezvoltarea locală:

```
$ mkvirtualenv mlconjug
$ cd mlconjug/
$ python setup.py develop
```

4. Creați o sucursală pentru dezvoltarea locală:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

   Acum puteți face schimbările la nivel local.

5. Când ați terminat de făcut modificări, verificați dacă modificările dvs. trec prin flake8 și testele, inclusiv testarea altor versiuni Python cu tox

```
$ flake8 mlconjug tests
$ python setup.py test or py.test
$ tox
```

   Pentru a obține flake8 și tox, pur și simplu instalați-le în virtualenv.

6. Comutați modificările și împingeți sucursala dvs. în GitHub

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Trimiteți o solicitare de tragere prin intermediul site-ului GitHub.

5.3 Instrucțiuni de solicitare trageți

Înainte de a trimite o solicitare de tragere, verificați dacă respectă aceste linii directoare:

1. Cererea de tragere ar trebui să includă teste.
2. Dacă cererea de trasare adaugă funcționalitate, documentele trebuie să fie actualizate. Puneți noua funcție într-o funcție cu un docstring și adăugați caracteristica în lista în README.rst.
3. The pull request should work for Python 3.3, 3.4, 3.5 and 3.6. Check https://travis-ci.org/SekouD/mlconjug/pull_requests and make sure that the tests pass for all supported Python versions.

5.4 Sfaturi

Pentru a rula un subset de teste

```
$ py.test tests.test_mlconjug
```

CAPITOLUL 6

Credite

6.1 Lead de dezvoltare

- SekouD <sekoud.python@gmail.com> GPG key ID: B51D1046EF63C50B

6.2 Contribuabili

- Sigla a fost creată de Zuur.

CAPITOLUL 7

Istorie

7.1 3.4 (2019-29-04)

- Fixed bug when verbs with no common roots with their conjugated form get their root inserted as a prefix.
- Added the method iterate() to the Verb Class as per @poolebu's feature request.
- Updated Dependencies.

7.2 3.3.2 (2019-06-04)

- Corrected bug with regular english verbs not being properly regulated. Thanks to @vectomon
- Updated Dependencies.

7.3 3.3.1 (2019-02-04)

- Corrected bug when updating dependencies to use scikit-learn v 0.20.2 and higher.
- Updated Dependencies.

7.4 3.3 (2019-04-03)

- Updated Dependencies to use scikit-learn v 0.20.2 and higher.
- Updated the pre-trained models to use scikit-learn v 0.20.2 and higher.

7.5 3.2.3 (2019-26-02)

- Updated Dependencies.
- Fixed bug which prevented the installation of the pre-trained models.

7.6 3.2.2 (2018-18-11)

- Updated Dependencies.

7.7 3.2.0 (2018-04-11)

- Updated Dependencies.

7.8 3.1.3 (2018-07-10)

- Updated Documentation.
- Added support for pipenv.
- Included tests and documentation in the package distribution.

7.9 3.1.2 (2018-06-27)

- Actualizați „Adnotați tip” la întreaga bibliotecă pentru conformitatea cu PEP-561.

7.10 3.1.1 (2018-06-26)

- Aplicația Minor Api (consultați documentația API)

7.11 3.1.0 (2018-06-24)

- Actualizate modelele de conjugare pentru spaniolă și portugheză.
- Modificări interne la formatul datelor verbice de la xml la json pentru o mai bună manipulare a caracterelor unicode.
- Noua clasă ConjugManager pentru a adăuga mai ușor limbi noi în mlconjug.
- Aplicația Minor Api (consultați documentația API)

7.12 3.0.1 (2018-06-22)

- **Actualizate toate modelele de predicție pre-instruire furnizate:**
  - Implementat un nou vectrorizer care extrage caracteristici mai semnificative.
  - Ca rezultat, performanța modelelor a trecut prin acoperiș în toate limbile.
  - Recall și Precision sunt aproape de 100%. Limba engleză fiind singura pentru a obține un scor perfect atât la Recall, cât și la Precizie.
- **Principalele modificări API:**
  - Am scos clasa EndingCustomVectorizer și am refactat funcționalitatea acesteia într-o funcție de nivel superior numită extract_verb_features ()
  - Modelul îmbunătățit nou furnizat este acum comprimat cu zip înainte de lansare, deoarece spațiul caracteristicilor a crescut atât de mult încât dimensiunea acestora le-a făcut imposibil de distribuit împreună cu pachetul.
  - Redenumit „Model.model" la „Model.pipeline"
  - Redenumit „DataSet.liste_verbes" și „DataSet.liste_templates" la „DataSet.verbs_list" și, respectiv, „DataSet.templates_list". (Scuza-mi limbajul ;-))
  - A adăugat atributele „predicted" și „confidence_score" la verbul de clasă.
  - Întregul pachet a fost tipărit verificat. Voi adăuga în curând niște stub-uri de tip mlconjug pe care să le tipărești.

7.13 2.1.11 (2018-06-21)

- **Actualizate toate modelele de predicție pre-instruire furnizate**
  - Conjugatorul francez are o precizie de aproximativ 99,94% în prezicerea clasei corecte de conjugare a unui verb francez. Aceasta este linia de bază deoarece am lucrat la ea de ceva timp acum.
  - Conjugatorul englez are o precizie de aproximativ 99,78% în prezicerea clasei corecte de conjugare a unui verb englez. Aceasta este una dintre cele mai mari îmbunătățiri de la versiunea 2.0.0
  - Conjugatorul spaniol are o precizie de aproximativ 99,65% în prezicerea clasei corecte de conjugare a unui verb spaniol. De asemenea, a înregistrat o îmbunătățire considerabilă de la versiunea 2.0.0
  - Conjugatorul român are o precizie de aproximativ 99,06% în prezicerea clasei corecte de conjugare a unui verb românesc. Acesta este cu mult câștigul mai mare. Am modificat vectorul pentru a ține mai bine cont de caracteristicile morfologice sau de verbele românești. (scorul anterior a fost de aproximativ 86%, așa că va fi frumos pentru prietenii noștri români să aibă un conjugator de încredere)
  - Conjugatorul portughez are o precizie de aproximativ 96,73% în predicția clasei corecte de conjugare a unui verb portughez.
  - Conjugatorul italian are o precizie de aproximativ 94,05% în prezicerea clasei corecte de conjugare a unui verb italian.

7.14 2.1.9 (2018-06-21)

- **Acum, conjugatorul adaugă informații suplimentare obiectului Verb returnat.**

  - Dacă verbul în cauză este deja în Verbiste, conjugarea pentru verb este extrasă direct din memorie.
  - Dacă verbul în cauză nu este cunoscut în Verbiste, clasa Conjugator stabilește acum atributul boolean «predicted» și scorul de încredere a atributului float la instanța obiectului Verb conjugator.conjugate (verb) revine.

- Adăugat *Adnotări de tip* la întreaga bibliotecă pentru robustețe și ușurință de scalare.
- Performanțele modelelor engleze și române s-au îmbunătățit semnificativ în ultimul timp. Cred ca, in cateva iteratii, ele vor fi pe masura cu modelul francez, care este cel mai performant in momentul in care am fost tuning parametrii pentru un caouple de ani acum. Nu atât de mult cu celelalte limbi, dar dacă actualizați în mod regulat veți vedea improvizații frumos în versiunea 2.2.
- Ameliorat localizarea programului.
- Acum, interfața cu utilizatorul de mlconjug este disponibilă în engleză, franceză, spaniolă, italiană, portugheză și română.
- „Toate documentațiile proiectului" au fost traduse în limbile acceptate.

7.15 2.1.5 (2018-06-15)

- Localizare adăugată.
- Acum, interfața cu utilizatorul de mlconjug este disponibilă în engleză, franceză, spaniolă, italiană, portugheză și română.

7.16 2.1.2 (2018-06-15)

- A fost adăugată detectarea verbelor invalide.

7.17 2.1.0 (2018-06-15)

- Actualizat toate modelele de limbă pentru compatibilitatea cu scikit-learn 0.19.1.

7.18 2.0.0 (2018-06-14)

- Include modelul de conjugare în limba engleză.
- Include modelul de conjugare spaniol.
- Include modelul de conjugare italian"
- Include modelul de conjugare portugheză.
- Include modelul de conjugare din România"

7.19 1.2.0 (2018-06-12)

- A fost refactat API-ul. Acum este nevoie de un singur conjugator de clasă pentru a interfața cu modulul.
- Include un model de conjugare francez îmbunătățit"

- Suport adăugat pentru mai multe limbi.

7.20 1.1.0 (2018-06-11)

- A fost refactat API-ul. Acum este nevoie de un singur conjugator de clasă pentru a interfața cu modulul.
- Include un model de conjugare francez îmbunătățit"

7.21 1.0.0 (2018-06-10)

- Prima versiune pe PyPI"

CAPITOLUL 8

Indici și tabele

- genindex
- modindex
- search

Indexul de Module Python

m

Index

Simboluri

C

D

E

G

I

M

P

S

T

V