ML Frameworks and tools

Scikit-learn:

Scikit-learn is probably the most useful library for machine learning in Python. The sklearn library contains a lot of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction.

Please note that sklearn is used to build machine learning models. It should not be used for reading the data, manipulating and summarizing it.

**Components of scikit-learn:**

Scikit-learn comes loaded with a lot of features. Few of them as follow:

* **Supervised learning algorithms:** Starting from Generalized linear models (e.g Linear Regression), Support Vector Machines (SVM), Decision Trees to Bayesian methods – all of them are part of scikit-learn toolbox. The spread of machine learning algorithms is one of the big reasons for the high usage of scikit-learn.
* **Cross-validation:** There are various methods to check the accuracy of supervised models on unseen data using sklearn.
* **Unsupervised learning algorithms:** There is a large spread of machine learning algorithms in the offering – starting from clustering, factor analysis, principal component analysis to unsupervised neural networks.
* **Feature extraction:** Scikit-learn for extracting features from images and text (e.g. Bag of words)

By using scikit-learn use pre-processing. LabelEncoder() to process the data, and train\_test\_split() to split the data set into test and train samples. We will also use a cheat sheet to help us decide which algorithms to use for the data set.

Eg.-

#import the necessary module

from sklearn import preprocessing

# create the Labelencoder object

le = preprocessing.LabelEncoder()

#convert the categorical columns into numeric

encoded\_value = le.fit\_transform(["paris", "paris", "tokyo", "amsterdam"])

print(encoded\_value)

spaCy

Spacy is an open-source software python library used in advanced natural language processing and machine learning. It will be used to build information extraction, natural language understanding systems, and to pre-process text for deep learning.

* Is spaCy or NLTK better?

spaCy has support for word vectors whereas NLTK does not. As spaCy uses the latest and best algorithms, its performance is usually good as compared to NLTK. As we can see below, in word tokenization and POS-tagging spaCy performs better, but in sentence tokenization, NLTK outperforms spaCy.

**Features:**

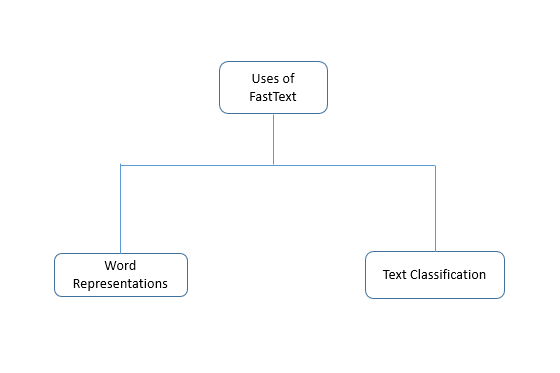
| **NAME** | **DESCRIPTION** |
| --- | --- |
| **Tokenization** | Segmenting text into words, punctuations marks etc. |
| **Part-of-speech** (POS) **Tagging** | Assigning word types to tokens, like verb or noun. |
| **Dependency Parsing** | Assigning syntactic dependency labels, describing the relations between individual tokens, like subject or object. |
| **Lemmatization** | Assigning the base forms of words. For example, the lemma of “was” is “be”, and the lemma of “rats” is “rat”. |
| **Sentence Boundary Detection** (SBD) | Finding and segmenting individual sentences. |
| **Named Entity Recognition** (NER) | Labelling named “real-world” objects, like persons, companies or locations. |
| **Entity Linking** (EL) | Disambiguating textual entities to unique identifiers in a knowledge base. |
| **Similarity** | Comparing words, text spans and documents and how similar they are to each other. |
| **Text Classification** | Assigning categories or labels to a whole document, or parts of a document. |
| **Rule-based Matching** | Finding sequences of tokens based on their texts and linguistic annotations, similar to regular expressions. |
| **Training** | Updating and improving a statistical model’s predictions. |
| **Serialization** | Saving objects to files or byte strings. |

**Pros and Cons of spaCy:**

* Pros: easy to use, very fast, ready for production, very customizable, widely used in deep learning research
* Cons: not customizable, internals are opaque, fewer NLP abstractions, not optimized for speed.

FastText:

FastText is a library created by the Facebook Research Team for **efficient learning of word representations** and **sentence classification**.



**fastText** is a library for learning of [word embeddings](https://en.wikipedia.org/wiki/Word_embedding) and text classification created by [Facebook](https://en.wikipedia.org/wiki/Facebook)'s AI Research (FAIR) lab. The model allows one to create an [unsupervised learning](https://en.wikipedia.org/wiki/Unsupervised_learning) or [supervised learning](https://en.wikipedia.org/wiki/Supervised_learning) algorithm for obtaining vector representations for words.

FastText differs in the sense that word vectors a.k.a word2vec treats every single word as the smallest unit whose vector representation is to be found but FastText assumes a word to be formed by a n-grams of character, for example, sunny is composed of [sun, sunn,sunny],[sunny,unny,nny]  etc, where n could range from 1 to the length of the word.

### Learning Word Representations

Words in their natural form cannot be used for any Machine Learning task in general. One way to use the words is to transform these words into some representations that capture some attributes of the word. It is analogous to describing a person as – [‘height’:5.10 ,’weight’:75, ‘colour’:’dusky’, etc.] where height, weight etc are the attributes of the person. Similarly, word representations capture some abstract attributes of words in the manner that similar words tend to have similar word representations. There are primarily two methods used to develop word vectors – Skipgram and CBOW.

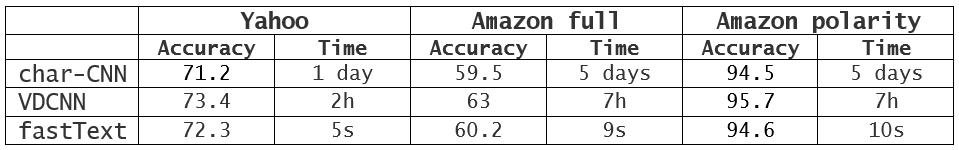
### Text Classification

As suggested by the name, text classification is tagging each document in the text with a particular class. Sentiment analysis and email classification are classic examples of text classification. In this era of technology, millions of digital documents are being generated each day. It would cost a huge amount of time as well as human efforts to categorise them in reasonable categories like spam and non-spam, important and unimportant and so on. Text classification techniques of NLP come here to our rescue.

## **Pros and Cons of FastText**

Like every library in development, it has its pros and cons. Let us state them explicitly.

### Pros

1. The library is surprisingly very fast in comparison to other methods for achieving the same accuracy. Here is the result published by the Facebook research team in support of the argument.
2. Sentence Vectors(supervised) can be easily computed.
3. fastText works better on small datasets in comparison to gensim.
4. fastText performs superior to gensim in terms of syntactic performance and fairs equally well in case of semantic performance.

### Cons

1. This is not a standalone library for NLP since it will require another library for the pre-processing steps.
2. Though, this library has a python implementation. It is not officially supported.