Notebook

February 25, 2019

```
In [1]: # import modules & set up logging
        import gensim, logging
        import smart_open, os
        logging.basicConfig(format='%(asctime)s : %(levelname)s : %(message)s', level=logging.I
        import datetime
        import pandas as pd
        import multiprocessing

        # fichier incltu dans le projet
        import save_notebook
```

```
D:\Outil\Anaconda\envs\majeure-ml-env\lib\site-packages\gensim\utils.py:1197: UserWarning: dete
  warnings.warn("detected Windows; aliasing chunkize to chunkize_serial")
```

1 DÃl'claration donnÃl'es

```
In [2]: now = str(datetime.datetime.now()).replace(" ","")
```

```
In [3]: word_embedding = "word2vec"
```

2 Prepare data

```
In [4]: filenames =  os.listdir("../wikipedia/data")
```

```
In [ ]: #CrÃl'Ãl' un fichier ou chaque ligne continent tout un fichier
        # path="../wikipedia/data"
        path="../wikipedia/data/"
        with open('./data/wikipedia_informatic.txt', 'w+',encoding="utf8" ) as out_file:

            for fname in filenames:
        #         print(fname)
                if "ipynb_checkpoints" in fname:
                    continue
                try:
                    with open(path + fname, encoding="utf8") as in_file:
                        out_file.write(in_file.read().replace("\n",""))
                except:
                    continue
```

```
In [ ]: # On lit et on tokenize le fichier
        with open('./data/wikipedia_informatic.txt', 'r', encoding="utf8") as f:
            wiki_vocab = f.readlines()
        wiki_vocab = [x.strip() for x in wiki_vocab]

        wiki_vocab_tokenized = []
        # for line in wiki_vocab:
        #     print(gensim.utils.simple_preprocess(line))
        # wiki_vocab_tokenized.append(gensim.utils.simple_preprocess(str(wiki_vocab)))
```

```
In [ ]: wiki_vocab_tokenized = gensim.utils.simple_preprocess(str(wiki_vocab))
```

3 Create model

```
In [ ]: # build vocabulary and train model
        model = gensim.models.Word2Vec(
            [wiki_vocab_tokenized],
            size=150,
            seed=1234,
            window=10,
            min_count=2,
            workers=multiprocessing.cpu_count())

        date_before_learning = datetime.datetime.now()
        model.train([wiki_vocab_tokenized], total_examples=len(wiki_vocab_tokenized), epochs=2
        time_training = datetime.datetime.now() - date_before_learning
```

```
In [ ]: model.save(str("model/" + now.replace(".","-").replace(":","-") + ".model"))
```

4 Test

```
In [ ]: result = model.wv.most_similar(positive="microsoft",topn=10)
```

```
In [ ]: print(result)
```

5 Save

```
In [ ]: import save_notebook
```

```
In [ ]: name_notebook_exported = save_notebook.save_notebook("word2vec_with_gensim.ipynb")
```

```
In [ ]: def write_result(word_embedding, time_training,name_notebook_exported, fname ):
            if not os.path.isfile(fname):
                f=open(fname, "a+")
                f.write("Nombre de mots;Model de word embedding;temps d'apprentissage;Notebook
                f.close
            f=open(fname, "a+")
            f.write("\n" + str( str(len(wiki_vocab_tokenized)) + ";" +word_embedding + ";"+ st
            f.close
```

```
In [ ]: write_result(word_embedding, time_training,name_notebook_exported, "resultats.csv" )
```