Trabalho 2 - NLP

Bruno Rodrigues Silva

[27]:
```
from nltk import word_tokenize, sent_tokenize, tokenize
import numpy as np
import nltk
import pandas as pd
from nltk.util import ngrams
stop = nltk.corpus.stopwords.words('portuguese')
from IPython.display import display, HTML
from nltk.lm import MLE
nltk.download('punkt')
from collections import Counter
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     /home/brunorosilva/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
```

[2]:
```
corpus = open('DomCasmurro.txt','r').read()
```

[259]:
```
class model():
    def __init__(self, corpus):
        self.corpus = corpus
        self.preprocess_done = False
        self.stop = nltk.corpus.stopwords.words('portuguese')
        self.lst_sents = []


    def _tokenizar(self, s):
        return tokenize.word_tokenize(s, language='portuguese')

    def _limpar(self, lista):
        return [i.lower() for i in lista if i.isalpha()]

    def _achatar(self, lista):
        return [i for sublista in lista for i in sublista]

    def _remover_pontuacao(self):
        self.pontos = ['...', ':', ';', '!', '?']
        for i in self.pontos:
            self.corpus = self.corpus.replace(i, '.')
```

```
    def _remover_barra_n(self):
        self.corpus_nl_removed = ""
        for line in self.corpus:
            line_nl_removed = line.replace("\n", " ")      #removes newlines
            self.corpus_nl_removed += line_nl_removed

    def _remover_pontuacao_e_barra_n(self):
        self.corpus_limpo = "".join([char for char in self.corpus_nl_removed if␣
↪char not in (self.pontos + ['\n'])])

    def _estatisticas_corpus(self):
        self.sents = sent_tokenize(self.corpus_limpo)
        self.words = word_tokenize(self.corpus_limpo)
        stats_df = pd.DataFrame({
            "Sentências":[len(self.sents)],
            "Palavras":[len(self.words)],
            "Média de palavras por sentença":[round(len(self.words)/len(self.
↪sents))],
            "Quantidade de palavras únicas":[len(set(self.words))],
        }, index=["Estatísticas"]).transpose()

        display(stats_df)


    def _criar_lst_sents(self):
        for sentence in self.sents:
            if len(sentence) > 0:
                self.lst_sents.append(self._limpar(self._tokenizar(sentence)))
        for i in range(len(self.lst_sents)):
            self.lst_sents[i] = ["<s>"] + self.lst_sents[i] + ["</s>"]

    def _remover_primeiros(self, n=10):
        self.lst_sents = self.lst_sents[10:]

    def preprocess(self):

        if self.preprocess_done == True:
            print("O preprocessamento já foi feito, você pode criar os modelos␣
↪diretamente")

        else:
            print("Começando o preprocessamento")

            self._remover_pontuacao()
            print("Pontuação Removida")
```

```
            self._remover_barra_n()
            print("Quebras de linhas removidas")
            self._remover_pontuacao_e_barra_n()
            print("Criação do corpus limpo")
            self._estatisticas_corpus()
            print("Estatísticas do corpus")
            self._criar_lst_sents()
            print("Criando lst sents")
            self._remover_primeiros()
            print("Removendo as 10 primeiras linhas (não fazem parte da obra)")

        self.preprocess_done = True

    def criar_ngrams(self, n_list=[1, 2, 3]):
        self.ngrams = {}

        for n in n_list:
            self.ngrams[str(n)+"gram"] = []

            for s in self.lst_sents:
                if s == "." and n==1:
                    pass
                else:
                    self.ngrams[str(n)+"gram"].append(list(ngrams(s, n)))

    def predict_text(self, limit=40, seed=42):
        self.predicts = {}
        for ngram in self.ngrams:

            pred = "<s> "
            model = MLE(int(ngram[0]))

            model.fit(self.ngrams[ngram], vocabulary_text=list(set(self.
↪words))+["<s>", "</s>", "\n"])
            model.fit(self.ngrams["1gram"])

            for p in model.generate(limit, text_seed=["<s>"], random_seed=seed):
                pred = pred+" "+p

                if p == "</s>":
                    break


            self.predicts[ngram]=pred
```

Instanciando um objeto Modelo e criando modelos para ngramas 1, 2 e 3 (padrão)

[260]:
```
m = model(corpus)
m.preprocess()
m.criar_ngrams()
```

```
Começando o preprocessamento
Pontuação Removida
Quebras de linhas removidas
Criação do corpus limpo
```

	Estatísticas
Sentências	5760
Palavras	83048
Média de palavras por sentença	14
Quantidade de palavras únicas	11196

```
Estatísticas do corpus
Criando lst sents
Removendo as 10 primeiras linhas (não fazem parte da obra)
```

[261]:
```
m.predict_text(seed=5)
```

[262]:
```
m.predicts
```

[262]:
```
{'1gram': '<s>  minha pae protecção um padecem tinha </s>',
 '2gram': '<s>  não se pintou a principio suppuz que lhe tirasse da terceira
</s>',
 '3gram': '<s>  minha mãe que tinha os seus </s>'}
```

[ ]:
```

```