Importando um corpus textual

Transcrição

Iniciaremos a construção do corretor ortográfico.

O primeiro passo será abrir o ambiente de desenvolvimento, e neste caso utilizaremos o *notebook* do **Google Colab** acessível neste **link** (https://colab.research.google.com/).

Se tivermos uma conta no **Gmail** já logada no *browser*, abriremos automaticamente uma janela com várias opções mostrando os notebooks já existentes. Ao final da lista, teremos o link "New Python 3 Notebook" que abrirá um novo documento `.ipynb` e o salvará no *drive*.

Clicaremos sobre o nome padrão automático e renomearemos esse notebook como `Corretor.ipynb`. Com isso, teremos o ambiente para desenvolver nosso modelo capaz de realizar a correção ortográfica.

Sempre quando queremos aprender novos conhecimentos, buscamos mais informações sobre o assunto em livros, vídeos, cursos, palestras, aulas e etc. Para que nosso modelo corrija as palavras, precisaremos primeiro ensiná-lo a escrever através de um **vocabulário** que cresce conforme aprende.

Logo, iremos precisar de uma **base de dados**.

Porém, como estamos trabalhando com o buscador do site da Alura, seria interessante também que essa base de conhecimento tivesse termos mais técnicos da área.

A base de dados que utilizaremos neste curso será construída com os próprios artigos do **Blog da Alura**, pois ensinaremos o nosso corretor a realizar correções específicas para o mundo técnico de desenvolvimento, como Java, programação orientada a objeto, *Data Science* e etc.

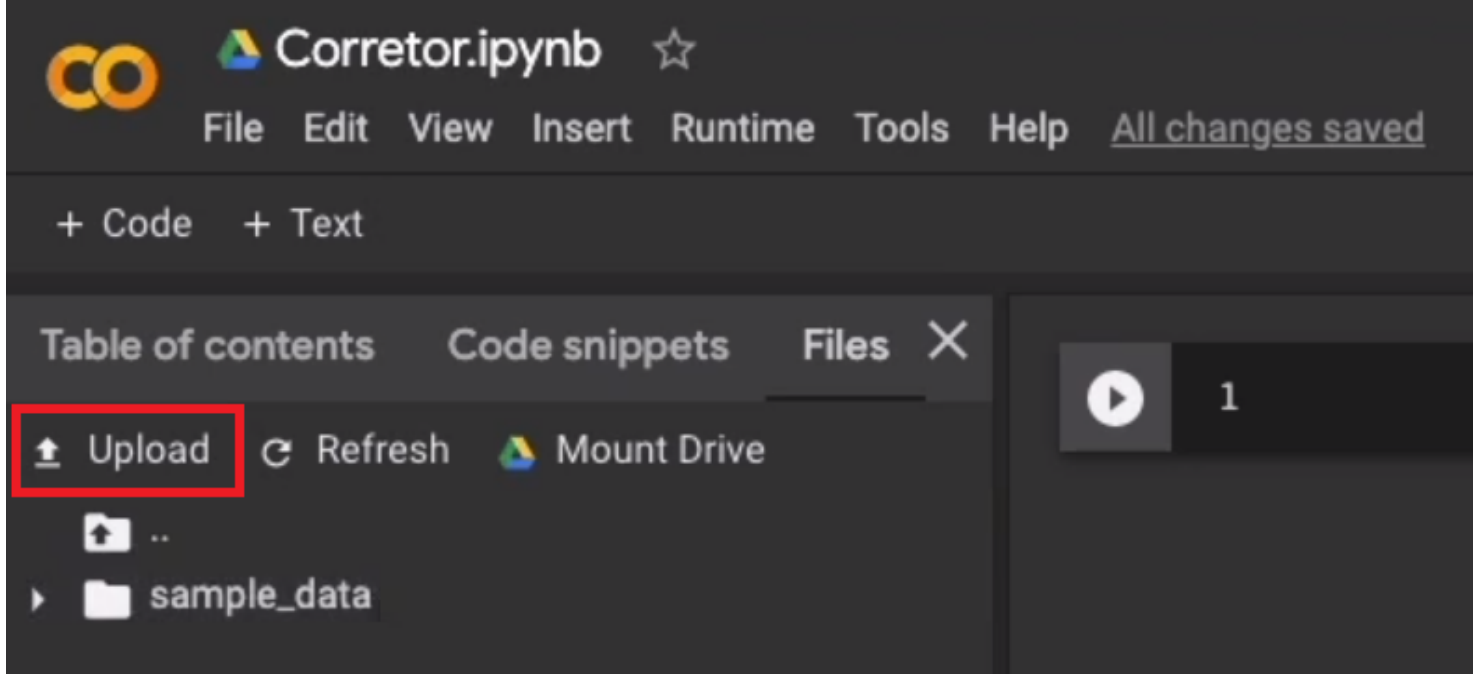
Nesta página de Artigos de Tecnologia e Negócios da Plataforma Alura (https://www.alura.com.br/artigos) teremos várias opções com diversos **artigos**. Como exemplo, abriremos o artigo sobre criação de formulários com Flutter (https://www.alura.com.br/artigos/criando-formulario-com-flutter).

Este é um artigo do instrutor **Alex Felipe** ensinando a criar um formulário utilizando **Flutter**, o qual possui um volume grande de texto que realmente nos interessa, além de alguns trechos de código e imagens.

Portanto, teremos um arquivo cheio de informações textuais do *blog*. Precisaremos importar esses dados para começarmos a analisar e trabalhar no nosso corretor. Para isso, os enviaremos para a máquina do Google.

No Google Colab, expandiremos o painel lateral clicando no ícone ">", e encontraremos diversas opções na barra superior. Dentre estas, clicaremos em "Files" e aguardaremos o processo de montagem da máquina.

Após isso, encontraremos os diretórios criados automaticamente e três opções de ações: "Upload", "Refresh" e "Mount Drive". Clicando sobre a primeira destas, poderemos enviar os dados para as máquinas do Google.



No passo "Preparando o Ambiente" (https://cursos.alura.com.br/course/nlp-corretor-ortografico/task/68762), encontraremos o link para download do arquivo (https://github.com/alura-cursos/corretor/archive/master.zip) da base de dados `artigos.txt` disponibilizada para este curso.

Com o documento já baixado e salvo no computador, o selecionaremos e faremos seu *upload*. Desta forma, acessaremos essa *database* a partir do meu notebook.

Quando trabalhamos em NLP, a nossa base de dados é conhecida como ***corpus***. Ou seja, é um corpo composto por com diversos textos, e cada um corresponde a um artigo do nosso blog, chamado de **documento**.

Logo, o corpus é um **conjunto de documentos** em Processamento Linguagem Natural, e no nosso caso, é composto pelos artigos do blog que formam a base de dados `artigos.txt` . Em seguida, a leremos no notebook.

Para lermos um arquivo textual em Python, usaremos o comando **`open()`**. Dentro dos parênteses, passaremos dois parâmetros; o primeiro será o nome do arquivo `artigos.txt` entre aspas, e o segundo será o que queremos fazer com este último, ou seja, queremos fazer a leitura.

Então passaremos o `"r"` de *read* como segundo parâmetro para lermos o arquivo. Ao executarmos, esperaremos um retorno de um texto como `string`.

```
open("artigos.txt", "r")
```

COPIAR CÓDIGO

Porém, essa linha de código retorna `TextIOWrapper` que não é o que esperávamos, pois ainda queremos uma `string`.

Para abrirmos o texto, utilizaremos o comando **`with`** antes de `open()`. Após essa função, escreveremos `as` e diremos que representaremos `f:` de *file*, ou "arquivo" em português.

```
with open("artigos.txt", "r") as f:
```

COPIAR CÓDIGO

Essa linha de código é bem parecida com um texto escrito em inglês mesmo.

Em seguida, passaremos o que queremos fazer com o `artigos.txt` aberto depois dos dois pontos, ou seja, queremos fazer a leitura.

Na linha seguinte, escreveremos `f` com **`.read()`** para lermos o arquivo. Também deveremos armazená-lo em uma nova variável chamada **`artigos`**. A imprimiremos com **`print()`** recebendo

`artigos[]`. Como nosso corpus é bastante grande, visualizaremos apenas os quinhentos primeiros caracteres inserindo `:500` dentro dos colchetes.

```
with open("artigos.txt", "r") as f:
    artigos = f.read()

print(artigos[:500])
```

COPIAR CÓDIGO

Como retorno, teremos um arquivo de texto e um tratamento que facilitará nosso trabalho.

A primeira palavra é "imagem" e representa a figura presente na página original de artigos. Onde havia trechos de código, substituímos pela palavra “Java”. Desta forma, ficaremos apenas com termos em português para criarmos o nosso corretor.

A seguir, descobriremos se esse arquivo é de fato um corpus interessante para essa construção; se há um número suficiente de palavras, ou se as repete muitas vezes.