DHENY R. FERNANDES

DATA ANALYTICS

DADOS GERADOS POR HUMANOS

AULA 03

SUMÁRIO

O QUE VEM POR AÍ?

Nesta aula, estudaremos o problema de classificação de texto. Selecionaremos uma base de texto, aplicaremos as técnicas de pré-processamento, vistas na última aula, e prepararemos os dados para inseri-los num algoritmo de machine learning. Por fim, veremos como estender o conceito de classificação de texto para análise de sentimentos e criaremos um modelo para analisar reviews de filmes.

HANDS ON

Seguindo o mesmo esquema da aula passada, mesclaremos conteúdo teórico e prático, usando o Jupyter notebook para desenvolver nossos códigos.

SAIBA MAIS

Introdução

Para começarmos a entender o problema de classificação de texto sem antes dar uma definição formal, vamos considerar alguns exemplos. Considere a imagem abaixo. Você a classificaria como um spam ou como uma mensagem legítima?

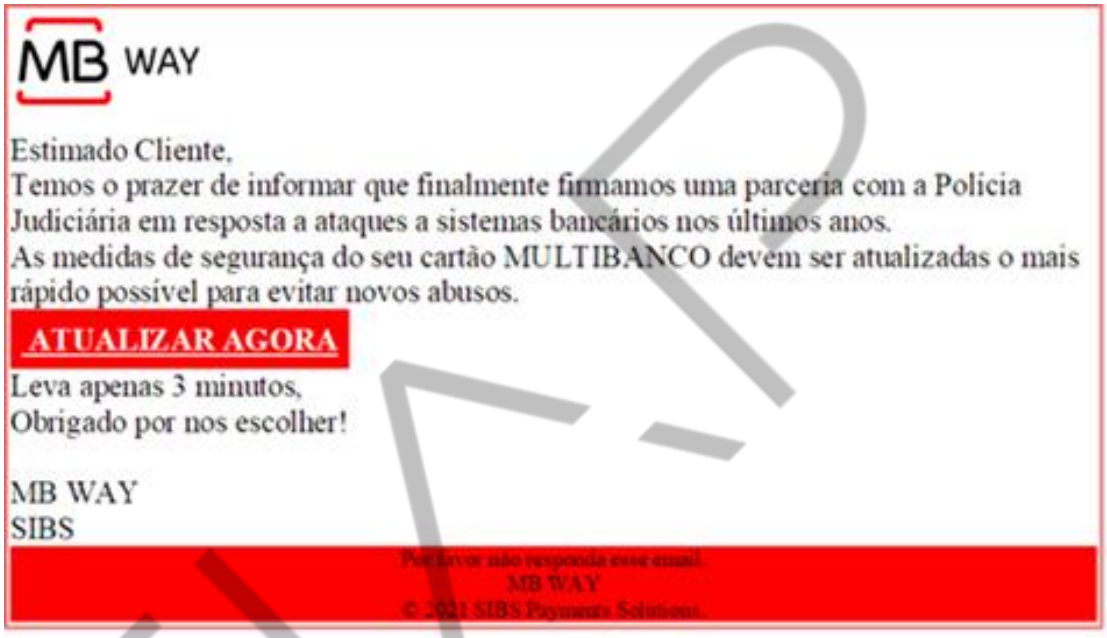

Figura 1: Spam
Fonte: https://www.policiajudiciaria.pt/alerta-de-phishing/ (s.d)

Olhando o texto escrito, é possível dizer que existe alguma característica que nos permita dizer se essa mensagem é fraudulenta ou não?

Vamos considerar mais um exemplo, retira de um artigo por Argamon et al, de 2003. Considere estas duas frases:

1. By 1925 present-day Vietnam was divided into three parts under French colonial rule. The southern region embracing Saigon and the Mekong delta was the colony of Cochin-China; the central area with its imperial capital at Hue was the protectorate of Annam…
2. Clara never failed to be astonished by the extraordinary felicity of her own name. She found it hard to trust herself to the mercy of fate, which had

> managed over the years to convert her greatest shame into one of her greatest assets…

Em tradução livre:

1. Em 1925, o Vietnã, como conhecemos hoje, estava dividido em três partes sob o domínio colonial francês. A região sul, abrangendo Saigon e o delta do Mekong, era a colônia da Cochinchina; a área central, com sua capital imperial em Huế, era o protetorado de Annam...
2. Clara nunca deixava de se surpreender com a extraordinária felicidade de seu próprio nome. Achava difícil confiar em si mesma à misericórdia do destino, que ao longo dos anos conseguiu transformar sua maior vergonha em um de seus maiores trunfos...

Analisando essas duas frases, você consegue identificar o gênero de quem as escreveu? Apesar de não ser uma tarefa trivial, é possível, com dados bons e em quantidade suficiente, treinar um classificador para identificar o gênero da pessoa autora de um texto.

Mais um exemplo: considere as seguintes críticas, escritas de uma maneira bem resumida:

- Incrivelmente desapontador.
- Cheio de personagens mirabolantes e uma sátira ricamente aplicada.
- O melhor filme de comédia já feito.
- Foi patético. A pior parte foi a cena dentro do saguão.

Ok, aqui é um pouco mais fácil, já que é algo que vemos quase que diariamente. Mas temos algumas questões importantes: como você conseguiu diferenciar uma crítica negativa de uma positiva? Que critérios você usou para estabelecer essa diferença? Foram as palavras, o contexto, ambos?

Apesar de descobrirmos o tipo de crítica logo após a ler a frase, precisamos pensar em como nosso cérebro processou as informações para chegar nessa conclusão, pois isso é o que devemos fazer, passo a passo, para que o modelo também consiga distinguir uma crítica positiva de uma negativa.

Por esses exemplos, podemos ver que existe uma grande quantidade de aplicações que envolvem classificação de textos. Aqui vão mais algumas:

- Atribuir categorias, tópicos ou gêneros a assuntos.
- Classificação de mensagens textuais.
- Identificação autoral.
- Identificação de língua escrita.
- Classificação de sentimento.
- Chatbots.

Embora o propósito e a aplicação da classificação de texto possam variar de domínio para domínio, o problema abstrato subjacente permanece o mesmo. Essa invariância do problema central e suas aplicações em uma infinidade de domínios torna a classificação de texto de longe a tarefa de NLP mais utilizada na indústria e a mais pesquisada na academia.

Em machine learning, a classificação é o problema de categorizar uma instância de dados em uma ou mais classes conhecidas. A classificação de texto é uma instância especial do problema de classificação, onde os dados de entrada são texto e o objetivo é categorizar o pedaço de texto em um ou mais grupos (chamados de classe) a partir de um conjunto de grupos predefinidos (Classes). O “texto” pode ter comprimento arbitrário: um caractere, uma palavra, uma frase, um parágrafo ou um documento completo.

Considerando o caso de reviews de filmes, o desafio de classificação de texto é aprender esta categorização a partir de uma coleção de exemplos para cada categoria e predizer a categoria para novos reviews, ainda não vistos.

Todo problema de classificação supervisionado pode ainda ser divido em três partes, baseado no número de classes envolvidas: binário, quando há apenas duas classes envolvidas; multiclasse, quando há mais de duas classes envolvidas (como quando quero classificar sentimentos em negativo, neutro ou positivo, por exemplo); e multilabel, em que um documento pode ter associado a ele uma ou mais classes, como num artigo de jornal, que pode tratar de política, negócios e assuntos jurídicos, por exemplo. Em nosso curso, trabalharemos com os dois primeiros casos.

Vamos, então, entender o problema de classificação de texto.

O problema de classificação de texto

Vamos definir formalmente o problema de classificação de texto.

Seja:

- $d \in X$ um documento, em que X é o espaço de documentos.
- $\mathbb{C} = \{c_1, c_2, \cdots, c_j\}$ um conjunto fixo de Classes (categorias ou rótulos).
- $\{(d_1, c_1), (d_2, c_2) \cdots, (d_m, c_m)\}$ um conjunto de treinamento de m documentos manualmente rotulados.

Usando um algoritmo de aprendizado, desejamos aprender uma função γ de classificação que mapeia documentos para classes:

- $\gamma = X \rightarrow \mathbb{C}$

Mas se $d \in X$ é um documento que pertence ao espaço de documentos, como representá-lo de forma que um computador consiga interpretá-lo? Como visto anteriormente, precisamos transformar esse documento em números.

O jeito como eu realizo essa transformação é denominado modelo de representação e esse modelo criado recebe o nome de vetor de características (nota: podemos usar features e descritores como sinônimos de características).

Inicialmente, vamos trabalhar com dois modelos de representação: **Bag-of-Words** e **TF-IDF**. São modelos mais simples, mas que oferecem um resultado muito satisfatório para várias aplicações. Inclusive, veremos um caso prático na aula 5. Vamos começar pelo Bag-of-words.

Importante notarmos que a representação que vimos na aula passada, em que colocamos 1 quando a palavra do dicionário está na frase e 0 quando contrário, é um tipo ainda mais simples, denominado One-Hot-encoding.

Bag-of-Words

O modelo Bag-of-Words (BoW) usa um vetor de contagens de palavras para representar um documento. Observe:

$x = [1,0,1,0,4,3,10,6,7,\cdots]$, em que x_j é a contagem da palavra j.

O tamanho de x é determinado pelo vocabulário $|\mathcal{V}|$, que é o conjunto de todas as possíveis palavras no vocabulário. Lembrando que o vocabulário é composto por todas as palavras distintas presentes no conjunto de documentos a serem classificados.

A intuição básica por trás disso é que ele assume que o texto pertencente a uma determinada classe no conjunto de dados é caracterizado por um conjunto único de palavras. Se dois trechos de texto tiverem quase as mesmas palavras, então eles pertencem ao mesmo conjunto (classe). Assim, ao analisar as palavras presentes em um trecho de texto, é possível identificar a classe (bag) a que ele pertence.

Por conta disso, o BoW somente inclui informação sobre a contagem de cada palavra e não sobre a ordem em que cada uma aparece. Logo, o contexto das frases é ignorado ao criar essa representação. Ainda assim, é surpreendentemente efetivo para classificação de texto.

A imagem a seguir mostra como funciona o BoW:

Figura 2: Representação usando BoW
Fonte: elaborado pelo autor (2023)

Esse tipo de representação é muito utilizado na maioria das aplicações de NLP. Aqui estão algumas vantagens de se usar:

1. Simples de entender e implementar.
2. Documentos que possuam as mesmas palavras terão vetores de representação próximos um do outro no espaço Euclidiano quando comparado com documentos com palavras completamente diferentes. Dessa maneira, o espaço vetorial resultante do BoW consegue capturar a similaridade semântica dos documentos.

Entretanto, possui também algumas desvantagens:

1. O tamanho do vetor de representação aumenta com o tamanho do vocabulário. Assim, esparsidade continua sendo um problema.
2. Ele não consegue capturar a similaridade entre diferentes palavras que possuam o mesmo significado. Por exemplo, considere três documentos: “Eu corro”, “Eu corri”, “Eu comi”. Todos eles estarão igualmente espaçados.
3. O BoW não possui uma maneira de lidar com palavras que estejam fora de seu vocabulário.
4. A informação da ordem que uma palavra aparece na frase é perdida com essa representação.

TF-IDF

Outro ponto que não comentamos a respeito dos modelos de representação vistos até o momento foi a importância da palavra num documento. Quando pensamos num texto, claramente algumas palavras possuem uma importância relativa maior do que outras. Para conseguir capturar isso, usamos outro tipo de representação, TF-IDF, que significa Term Frequency – Inverse Document Frequency.

O conceito de frequência de termo (TF) calcula a proporção de um termo num documento em relação ao número total de termos nesse documento. Entretanto, um problema com pontuar frequências de palavras é que palavras muito frequentes começam a dominar no documento (pontuação alta), mas podem não conter muita “informação de conteúdo” para o modelo quando comparadas a palavras raras que pertençam a domínios específicos. Assim, introduzimos um mecanismo para atenuar

o efeito de termos que ocorrem muito nos dados, para tornar significativo a determinação de sua relevância. Esse mecanismo é o IDF.

A intuição por trás do TF-IDF é a seguinte: se uma palavra w aparece muitas vezes em um documento d_i, mas não ocorre muito nos demais documentos d_j do corpus, então a palavra w deve ser de grande importância para o documento d_i. A importância de w deve aumentar proporcionalmente à sua frequência em d_i, mas, ao mesmo tempo, sua importância deve diminuir proporcionalmente a frequência da palavra em outros documentos d_j do corpus. Matematicamente, isso é capturado usando duas quantidades: TF e IDF. Os dois são então combinados, para chegar ao score TF-IDF.

TF mede a frequência com que um termo ou palavra ocorre em um determinado documento. Como diferentes documentos no corpus podem ter comprimentos diferentes, um termo pode ocorrer com mais frequência em um documento mais longo do que em um documento mais curto. Para normalizar essas contagens, dividimos o número de ocorrências pelo comprimento do documento. Assim, podemos definir TF da seguinte maneira:

$$tf = \frac{f_{t,d}}{\sum T, d}$$

IDF mede a importância do termo em um corpus. No cálculo do TF, todos os termos recebem igual importância (ponderação). No entanto, é um fato bem conhecido que palavras como “é”, “são”, “sou”, etc., não são importantes, embora ocorram com frequência. Para dar conta de tais casos, o IDF pondera para baixo os termos que são muito comuns em um corpus e pondera para cima os termos raros. O IDF é calculado da seguinte forma:

$$idf = log_e \frac{N}{n_t}$$

Por fim, multiplicamos os dois termos, obtendo o score final:

$$TFIDF = tf * idf$$

Em que:

t: termo analisado

d: documento analisado

T: conjunto de termos presente no documento d

N: número total de documentos

n_t: número de documentos que contém o termo t

Semelhante ao BoW, podemos usar os vetores TF-IDF para calcular a similaridade entre dois textos, usando uma medida de similaridade como distância euclidiana ou similaridade de cosseno. TF-IDF é uma representação comumente usada em aplicações como recuperação de informação e classificação de texto. No entanto, apesar do TF-IDF ser melhor que os métodos de vetorização que vimos anteriormente em termos de captura de semelhanças entre palavras, ele ainda sofre com a maldição da alta dimensionalidade. Veremos como amenizar esse problema na aula seguinte.

PRÁTICA

Agora, vamos analisar o código e aplicar os conhecimentos obtidos em um problema de classificação de texto.

ANÁLISE DE SENTIMENTOS

Análise de sentimentos é uma aplicação de classificação de textos. Geralmente, separamos os sentimentos em três classes: negativo, neutro e positivo. Como exercício de aplicação, temos uma base contendo tweets e nosso objetivo será criar um classificador que diga se o tweet analisado tem sentimento negativo, neutro ou positivo.

O Jupyter notebook desse exercício contém as orientações necessárias para a construção desse classificador. Hora de colocar em prática o que você viu até o momento.

O QUE VOCÊ VIU NESTA AULA?

Nesta aula, apresentamos o problema de classificação de texto, dois tipos de representação vetorial muito usados e um estudo de caso onde aplicamos o conhecimento obtido para classificar o sentimento de tweets.

O que achou do conteúdo? Conte-nos no Discord! Estamos disponíveis na comunidade para fazer networking, tirar dúvidas, enviar avisos e muito mais. Participe!

REFERÊNCIAS

ARGAMON, S et al. **Gender, Genre, and Writing Style in Formal Written Texts**. 23 (3), pp. 321–346. 2003.

BIRD, S; et al. **Natural Language Processing with Python**. Sebastopol: O'Reilly Media, 2009.

MIRANDA, O. **Normalização Unicode em Python.** Disponível em: <https://www.otaviomiranda.com.br/2020/normalizacao-unicode-em-python/>. Acesso em: 12 set. 2023.

VAJJALA, S; et al. **Practical Natural Languagem Processing: A Comprehensive Guide to Building Real-World NLP Systems**. Sebastapol: O'Reilly Media, 2020.

PALAVRAS-CHAVE

Natural Language Processing. Pré-processamento. NLTK.