Avaliação de Mensagens de Spam

Leonardo Fiedler
Leonardo.96.fiedler@gmail.com

1. Introdução

Mensagens de spam, segundo Viamonte, Silva e Macedo (2020), é originado do termo "*Sending and Posting Advertisement in Mass*" (Tráfego de publicidade em massa) ou "*Stupid Pointless Annoying Messages*" (mensagem de propósito irritante) e que são mensagens que tem por objetivo perturbar a navegação e o acesso a informação com algum conteúdo indesejado.

Além das mensagens de propaganda, o spam pode ser utilizado para aplicar golpes, efetuar estelionato, disseminar correntes e até propagar programas maliciosos. (VIAMONTE; SILVA; MACEDO, 2020).

Por conta disso, existem algumas ferramentas que tratam do assunto, buscando prevenir o usuário final de receber mensagens indesejadas e até mesmo cair em golpes. Um dos softwares conhecidos de mercado é chamado de SPAMfighter (Windows e Mac OS) que pode ser incorporado a ferramentas de e-mail. Outra ferramenta com esta mesma finalidade é o SpamSieve, este por sua vez funcionando com o Mac OS.

O objetivo deste trabalho está dividido em duas etapas, a primeira está voltada para visualização e análise dos dados, a partir de gráficos, filtros e agrupamentos. Já a segunda etapa concentra-se em criar um método capaz de classificar automaticamente as mensagens registradas (entre comum e spam).

2. Metodologia

A base de dados de entrada, consiste em um arquivo com extensão CSV (*comma-separated-values*), cujos valores são separados por vírgula, o qual possui as informações: mensagem completa, quantidade de ocorrências de cada uma das palavras, data, contagem de palavras e um indicativo se a mensagem é spam ou não. Este conjunto de dados já está filtrado e normalizado.

Para o correto funcionamento da base de entrada, é necessário efetuar um processo de transformação de dados e criação de colunas auxiliares para facilitar a filtragem e o agrupamento. A coluna "*IsSpam*", o qual possui valores ("*yes*" e "*no*") deve ser convertida para valores 1 e 0. Já a coluna "*Date*", pode ser separada em 3 colunas auxiliares, sendo elas: dia, mês e ano.

A linguagem adotada em todo o trabalho é o Python, onde algumas bibliotecas clássicas da linguagem também são utilizadas, como: *Pandas* (extração de informações do arquivo *CSV*) e *MatPlotLib* (geração de gráficos). Na classificação, a biblioteca *Sklearn* (*machine learning*) é escolhida. A biblioteca *Wordcloud* é utilizada somente para gerar a nuvem de palavras.

Ambas as etapas devem funcionar como CLI (*Command-line interface*), ou seja, executam em um terminal e é possível efetuar a passagem de parâmetros para a aplicação, como por exemplo, escolher qual algoritmo deseja-se executar.

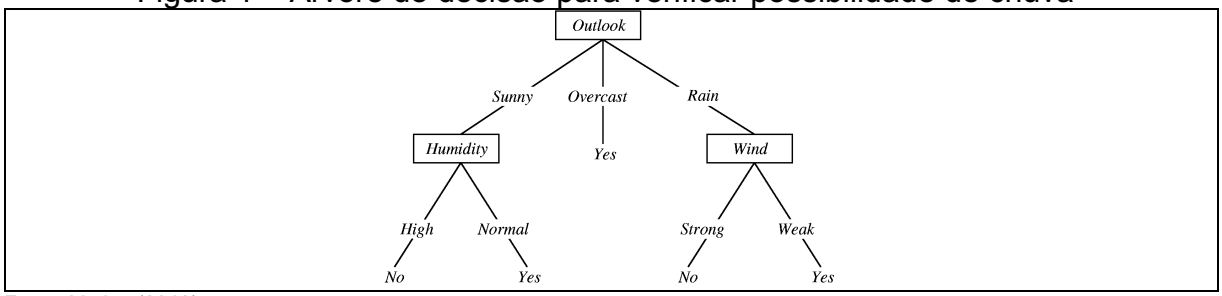
Na primeira etapa, é possível visualizar informações como: gráfico de frequência de palavras, nuvem de palavras, quantidade de mensagens comuns e spam por mês, dados estatísticos (máximo, mínimo, média, mediana, desvio padrão e variância de palavras por mês) e a sequência de dias do mês que possuem a maior quantidade seguidas de palavras comuns.

A segunda etapa, antes da utilização dos algoritmos, as informações são separadas em duas bases distintas, sendo os dados divididos em: 70% para treino e

30% para validações. Além disso, as entradas correspondentes são divididas em “X” (o qual representa os valores de entrada), que consiste na quantidade de cada palavra em cada mensagem e o “y” (o qual representa o rótulo de saída), este se utiliza da coluna “*IsSpam*” com os valores 0 e 1.

O primeiro algoritmo utilizado é chamado de *Decision Tree* Classifier (classificador de árvore de decisão), o qual segundo Yadav (2018) é uma estrutura semelhante a um fluxograma que consiste em cada nó interno representar um teste de uma característica, enquanto cada folha representa uma classe de saída. Na Figura 1 é possível ver um exemplo de uma árvore de decisão para verificar a possibilidade de clima chuvoso.

Figura 1 – Árvore de decisão para verificar possibilidade de chuva



Fonte: Yadav (2018).

Na árvore da Figura 1, é possível verificar que a possibilidade de chuva é influenciada pelos estados de sol, humidade e vento.

O segundo algoritmo utilizado é o *Random Forest Classifier* (classificador aleatório de floresta), segundo Yiu (2019) consiste em um grande número de árvores de decisão individuais que operam como um conjunto, sendo que cada árvore individual efetua a sua predição, o valor de classe considerado é o que mais se repete entre as árvores.

3. Resultados

Conforme a descrição do problema e a metodologia adotada, os resultados foram divididos em duas etapas, sendo a primeira voltada para análise e visualização de informações, enquanto a segunda para classificar as mensagens automaticamente.

A primeira etapa possui, dentre seus resultados, a nuvem e a frequência de palavras, a variação de mensagens comuns e de spam por mês, os dados estatísticos agregadores por mês e a maior sequência de mensagens que não possuem spam por mês.

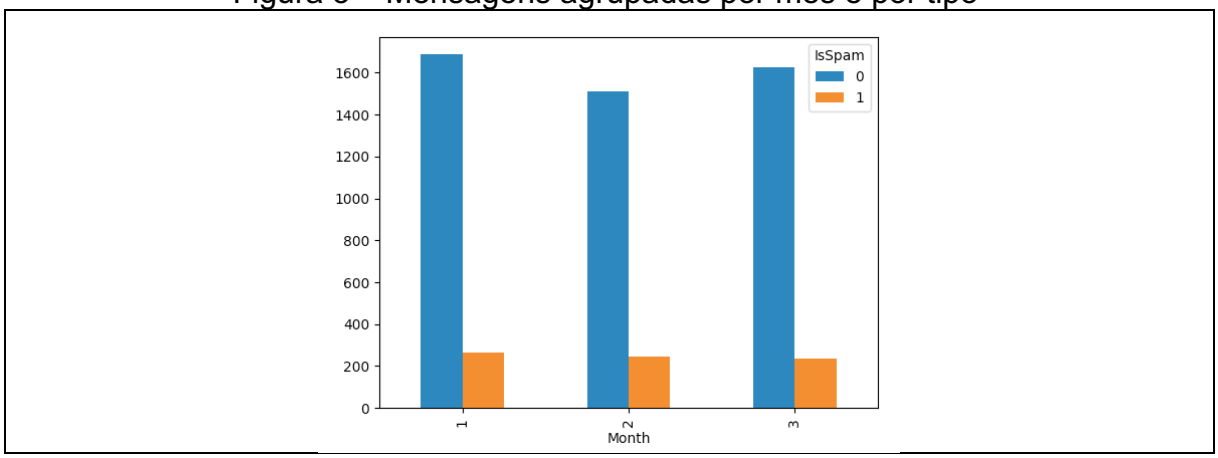
A frequência de palavras demonstra a quantidade de palavras ao longo de todos os meses, além de dar uma ideia da variação das palavras na base. Já a nuvem de palavras, busca o mesmo contraste, mas com outra organização (Figura 2).

Figura 2 – Nuvem de Palavras

Fonte: Elaborado pelo autor.

Conforme visualizado acima, as palavras que mais se destacam são: *call* (581 vezes), *now* (479 vezes), *can* (405 vezes), *get* (390 vezes) e *will* (383 vezes). Ao classificar as mensagens por mês e por tipo (spam ou comum) é possível ver que a variação de cada um dos conjuntos é baixa, conforme a Figura 3.

Figura 3 – Mensagens agrupadas por mês e por tipo



Fonte: Elaborado pelo autor.

Para analisar melhor as informações de forma sumarizada, foram adicionados alguns cálculos agregadores, agrupados por mês e que demonstram indicadores da variável "*Word_Count*", conforme Tabela 1.

Tabela 1 – Cálculos Agregadores

Mês	Max	Min	Média	Mediana	Desvio	Variância
1	190	2	16,34	13,0	12,52	157,68
2	100	2	16,03	13,0	11,04	121,94
3	115	2	16,29	12,0	11,58	134,01

Fonte: Elaborado pelo autor.

A tabela anterior pode ser interpretada como: no mês 1, a mensagem com a quantidade máxima de palavras foi de 190, enquanto a mínima foi de 2. A média das mensagens do mês 1 gira em torno de 16,34 palavras por mensagem, enquanto a mediana é 13,0. Já o desvio padrão é de 12,52 e a variância é de 157,68.

Na primeira etapa ainda é possível visualizar a quantidade de mensagens comuns seguidas, por mês. A Tabela 2 demonstra os valores obtidos.

Tabela 2 – Quantidade de mensagens comuns seguidas por mês

Mês	Quantidade	Dia do mês
1	31	26
2	39	4
3	46	31

Fonte: Elaborado pelo autor.

A segunda etapa, ao executar os dois algoritmos, foram extraídas algumas métricas, como por exemplo: erro absoluto médio, acurácia, matriz de confusão, precisão, *recal* e F1, podendo ser visualizadas na Tabela 3.

Tabela 3 – Execução de Algoritmos de Classificação

Algoritmo	Acurácia	Precisão	Recal	F1
Decision Tree	0,95	0,8	0,84	0,82
Random Forest	0,96	0,84	0,86	0,85

Fonte: Elaborado pelo autor.

Baseado nos resultados apresentados na tabela 3, pode-se afirmar que em ambos os casos foi possível efetuar a correta classificação e o algoritmo que obteve o melhor resultado foi o *Random Forest Classifier*, com acurácia de 96% e precisão de 84%.

4. Conclusão

Ao realizar o desenvolvimento do trabalho, é possível observar que a massa de dados ainda é pequena, para garantir que os resultados aqui obtidos possam ser assumidos como reais (ser executados em produção). Além disso, apesar de serem feitas as divisões dos dados, não é garantido um valor mínimo para as mensagens de spam, bem como as mesmas não são categorizadas, o que pode gerar algum tipo de BIAS.

Os algoritmos utilizados são ambos com base em árvores, o ideal é que outros tipos de algoritmos sejam utilizados para verificar se há realmente correlação nos dados ou se há alguma tendência dos dados da base. Algoritmos como *K-Means*, poderiam ser futuramente utilizados para classificar as mensagens de spam em diferentes tipos.

O resultado do experimento é positivo pois é possível ver que há um padrão claro nas mensagens de spam e é possível montar um algoritmo capaz de classificar as mensagens em spam ou comum. Em ambas as execuções, a acurácia ficou acima de 90%, o que corrobora com a tese acima afirmada.

5. Referências

VIAMONTE, Guilherme Avelino; SILVA, Kim Kaznowski da; MACEDO, Rodrigo de Jesus. SPAM. Disponível em: https://www.gta.ufrj.br/grad/15_1/spam/. Acesso em: 09 mar. 2020.

YADAV, Prince. Decision Tree in Machine Learning. 2018. Disponível em: https://towardsdatascience.com/decision-tree-in-machine-learning-e380942a4c96. Acesso em: 09 mar. 2020.

YIU, Tony. Understanding Random Forest: how the algorithm works and why it is so effective. How the Algorithm Works and Why it Is So Effective. 2019. Disponível em: https://towardsdatascience.com/understanding-random-forest-58381e0602d2. Acesso em: 09 mar. 2020.