ANÁLISE DE SENTIMENTOS EM AVALIAÇÕES DE NPS

RICARDO SILVEIRA MARTINS DE MELLO

Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS), Porto Alegre, Rio Grande do Sul

ricardosmdemello@gmail.com

RESUMO: Este artigo tem por finalidade, realizar a análise de sentimentos baseado nos comentários de avaliações de ***NPS* (*Net Promoter Score*)** da empresa *Safeweb Segurança da Informação Ltda*; No objetivo de descoberta se no momento de avaliação o sentimento era ***positivo*** ou ***negativo***. Para isso foram utilizadas técnicas de ciências de dados, bem como o treinamento de uma ***rede neural recorrente LSTM*** e um algoritmo de *machine learning* ***MultinomialNB* (*Naive Bayes*)**, utilizando os dados do ***imdb pt-br*** "como base de conhecimento auxiliar para treinamento"; Seguindo com a avaliação dos resultados apresentados após o treinamento da *rede neural* e do algoritmo de *classificação*, avaliando entre os dois modelos qual obteve o melhor resultado com as avaliações do ***NPS***. E finalizando com a análise dos resultados apresentados, com os respectivos sentimentos.

Palavras-chave: *NPS, Net Promoter Score, imdb pt-br, LSTM, rede neural, Machine Learning, MultinomialNB, Naive Bayes, Python, Análise de sentimento, avaliação, resultado, comentários, ciência de dados*.

SENTIMENT ANALYSIS IN NPS EVALUATIONS

ABSTRACT: This article has the purpose of performing a sentiment analysis based on the comments made in the ***NPS (Net Promoter Score)*** evaluation of the company *Safeweb Information Security Ltd*; In order to find out whether at the time of the evaluation the sentiment was positive or negative. For this, data science techniques will be used, as well as the training of a recurrent ***neural network LSTM*** and a machine learning algorithm ***MultinomialNB* (*Naive Bayes)***, using the "***imdb pt-br*** data as an auxiliary knowledge base for training"; Followed by the evaluation of the results presented after the training of the *neural network* and the *classification* algorithm, evaluating between the two models which which obtained the best result with the ***NPS*** evaluations. Finally, an ending with the analysis of the results are presented, with their respective *sentiments*.

Keywords: *NPS, Net Promoter Score, imdb pt-br, LSTM, neural network, Machine Learning, MultinomialNB, Naive Bayes, Python, Sentiment analysis, rating, score, feedback, data science*.

INTRODUÇÃO : Com a expansão da tecnologia e sua constante evolução ao longo do tempo, acabaram por fazer com que as empresas se informatizarem, além de introduzir seus negócios de forma gradual de forma virtual através da internet e com isso, surge então o grande desafio dessas empresas onde além de existir a necessidade de manter o negócio como um todo, também existe a necessidade de manter o negócio online, além de manter o nível de satisfação dos usuários e consumidores, a fim de manter e expandir o produto. Pensando nisso, surge então em 2003 a metodologia de satisfação do cliente (***Net Promoter Score***) que foi apresentada por *Fred Reichheld* até então consultor da *Bain & Company*. O ***Net Promoter Score*** (*NPS*), tem por finalidade captar o nível de satisfação e o nível de fidelidade dos clientes com o sistema, através de avaliações, tanto de forma textuais como notas numéricas, a fim de entender as necessidades do cliente diante do produto, se o cliente está satisfeito ou não e se o mesmo é um potencial para indicação e captação de novos clientes com o intuito de expansão do negócio, ou sistema e com isso, acionar o plano de ação para atender as expectativas do cliente, além disso com o uso do ***NPS*** também é possível que seja realizadas tarefas de benchmark, já que é possível que o gestor, ou responsável analisar os

indicadores de satisfação de empresas concorrentes, a fim de entender seu posicionamento perante o mercado.

O presente trabalho tem como objetivo demonstrar e entender, através da análise de sentimentos se os clientes da empresa *Safeweb Segurança e Informação LTDA* estavam satisfeitos ou insatisfeitos ao avaliar o produto, através dos comentários de avaliações dos dados de *NPS* coletados, para isso foram treinados dois algoritmos para a classificação, um algoritmo de machine learning ***MultinomialNB*** e uma rede ***neural recorrente LSTM,*** para realizar a classificação das avaliações de *NPS* apresentadas entre os sentimentos "positivo e negativo", com base nas respectivas avaliações, partindo das análises quantitativas, ou seja, as notas de satisfação e das análises qualitativas, representadas pelos comentários atrelados a nota dada. Para o treinamento dos algoritmos foi utilizado o ***imdb pt-br*** "dataset com dados rotulados, entre positivo e negativo" como base auxiliar para o treinamento dos algoritmos. Ao final será feito uma análise com o objetivo de avaliar qual algoritmo obteve os melhores resultados em suas classificações.

DADOS UTILIZADOS E ALGORITMOS: Durante a realização deste trabalho, foi utilizado uma base de dados ***imdb pt-br*** para o treinamento do algoritmo de machine learning e rede neural, e outra base ***NPS*** para a classificação:

- **Avaliações de NPS (*Net Promoter Score*)** "Período avaliado": Avaliação de satisfação do usuário com a plataforma; Fonte: Safeweb Segurança da Informação LTDA.
- **IMDB PT-BR**: Dataset popularmente utilizado para treinar modelos de classificação de textos, entre "**positivos**" e "**negativos**" através de testes. Fonte: Kaggle
- **MultinomialNB (*Naive Bayes*)**: Algoritmo de classificação probabilística, baseado no teorema de Bayes, que consiste em calcular um conjunto de probabilidades a quantidade de combinações de valores de um conjunto de dados e sua frequência.
- **Rede neural recorrente LTSM**: Algoritmo de rede neural recorrente (RNN).

A SOLUÇÃO: Para realizar a execução deste trabalho foram realizadas uma sequência de etapas, a fim de alcançar o melhor resultado dos testes efetuados, para isso primeiramente foi realizado a leitura dos dados de ***NPS*** que estavam dispostos em tabelas no formato *HTML,* transformando os dados lidos em um dataset das informações dispostas; Após a criação do dataset dos dados de ***NPS***, foi realizado o tratamento dos dados, bem como tratamento dos nomes de colunas do dataset na finalidade de padronizar as informações, ao final do tratamento foi realizado a contagem do total de comentários dispostos no dataset, onde pode-se obter o resultado de 7.521 resultados não nulos de um total de 15.970 comentários, em sequência também foi realizado o mesmo procedimento, porém com os dados dispostos através de um arquivo **.xlsx**, onde foram executados os mesmos tratamentos, obtendo ao final um resultado total de 24.490 resultados, sendo eles 15.970 resultados não nulos e finalizando com o agrupamento dos datasets tanto dos que foram gerados em *HTML*, quanto aos que foram gerados através do *XLSX*, transformando-os em um único dataset contendo 40.460 resultados; O dataset final, tem como informações o identificador da requisição, sua nota quantitativa, o comentário e o nome do responsável, sendo também realizado o tratamento para as requisições em duplicidade resultando em um total de 29.963 registros, sendo 10.325 registros com comentários registrados no dataset que será utilizado para a análise.

Para a segunda etapa, foi realizado o preparo do dataset ***imdb pt-br*** que contém comentários rotulados entre **positivos** e **negativos**, sendo esse dataset utilizado posteriormente para treinar o modelo de *machine learninng* ***MultinomialNB*** **(*Naive Bayes*)** e para o treinamento da ***rede neural recorrente LTSM***, para que os mesmos possam identificar e posteriormente classificar quando o resultado apresentado for **negativo** ou **positivo**, o dataset do imdb pt-br possui um total de 49.459 registros já rotulados. Seguindo com o preparo do dataset, foi-se removido todos os dados que não forem disponibilizados em *pt-br* para evitar que os

algoritmos possam interpretar e classificar de forma incorreta, já que os dados de ***NPS*** estão dispostos em *pt-br*, feito isso, realizou-se a conversão dos dados da coluna de "*sentimentos*" para valores booleanos, sendo "***positivo***" convertido para **1** e "***negativo***" convertido para **0** e por fim renomeando as colunas restantes.

Durante a terceira etapa deste trabalho, foi desenvolvido o modelo de treinamento utilizando o algoritmo ***MultinomialNB* (*Naive Bayes*)**, onde logo após a importação das bibliotecas necessárias, foram importados os dados do dataset ***imdb pt-br*** verificando a quantidade de palavras contidas no *dataset*, também foi necessário validar o *dataset* para verificar todas as classes diferentes entre **positivo** e **negativo**, onde foram realizadas as divisões dos dados, sendo 50% dos textos para testes e *50%* dos textos para o treinamento do modelo. Após realizar a divisão entre *teste* e *treinamento*, com o uso do ***pandas*** foram criados um *dataframe* de *teste* e um *dataframe* de *treinamento* do modelo, após estes passos foi desenvolvida uma classe utilizando o ***MultinomialNB***, onde nesta classe os dados do *dataset* foram convertidos para números e logo em seguida realizando a aprendizagem do algoritmo; ao finalizar o treinamento do algoritmo os resultados contendo textos novos foram devolvidos para o algoritmo e finalizando com a predição dos textos encontrados. Para o teste do algoritmo, foi gerado um conjunto de frases aleatórias a fim de validar a funcionalidade e o resultado emitido pelo algoritmo, ou seja, quando o teste foi executado, o algoritmo validou todas as frases contidas no conjunto e efetuou a predição se a frase que estava sendo analisada naquele momento tem seu teor **negativo** ou **positivo**, por fim retornando o conjunto com a avaliação dos resultados encontrados entre **1** e **0** por cada frase tendo uma *precisão* de *0.937*8, um *recall* de *0.8881* e seu *fscore* em *0.9123* , também utilizou-se de 40% dos dados ***imdb pt-br*** para a validação do modelo, tendo como resultado uma *precisão* de *0.9007*, um *recall* de *0.8542* e um *fscore* de *0.8768*, finalizando com a acurácia de *0.9143*. Seguindo logo após o treinamento do modelo, foi utilizado o *dataset* referente aos ***NPS*** que já foram tratados, conforme informado na primeira etapa, os dados referentes aos comentários contidos no *dataset* do ***NPS*** foram convertidos em uma lista de comentários e em seguida passados para o modelo ***MultinomialNB*** treinado, para que o mesmo possa avaliar os comentários e predizer quais comentários foram **positivos** e quais foram **negativos**, sendo possível avaliar um total de *10.325* resultados contidos no *dataset*, ao encontrar os resultados devolvidos pelo *modelo treinado*, os dados em questão foram devolvidos em uma coluna chamada ***MultinomialNB*** dentro do *dataset* de ***NPS***, conforme dispostos nos *dados 1* e *dados 2*.

Scikit Naive Bayes

Dados 1: ***Dados de validação do imdb pt-br, métricas de avaliação:*** precision: *0.937*8 recall: *0.8881* *fscore*: *0.9123*

Dados 2: ***Foram usados 40% de dados aleatórios do imdb pt-br, métricas de avaliação:*** precision: *0.9007* recall: *0.8542* *fscore*: *0.8768*

Acurácia: *0.9143*

Após a finalização da terceira parte, deu-se início a quarta e última parte deste trabalho, para isso, foi utilizado o ***Keras*** para criar a uma ***rede neural recorrente LTSM***. Primeiramente foi necessário validar o *dataset* ***imdb pt-b***r para verificar todas as classes diferentes entre **positivo** e **negativo**, onde foi realizado a divisão dos dados, sendo 50% dos textos para testes e *50%* para o treinamento do modelo. Após realizar a divisão entre *teste* e *treinamento*, com o uso do ***pandas*** foram criados um *dataframe* de *teste* e um *dataframe* de *treinamento* do modelo, por fim utilizou-se de 30% dos dados ***imdb pt-br*** para a validação do modelo, tendo como resultado *24.730* dados para *teste*, *24.729* dados *treinamento* e *14.838* dados que serão utilizados para realizar a *validação*, para realizar o treinamento também foi necessário a criação de uma classe para realizar a ***tokenização*** dos textos, ou seja, transformar todos os textos em vetores inteiros, onde os dados utilizados para o ***treinamento***, ***teste*** e ***validação*** foram transformados em ***tokens***, finalizando em uma amostra em formato de *histograma* presente na *Imagem 1*, representada abaixo.

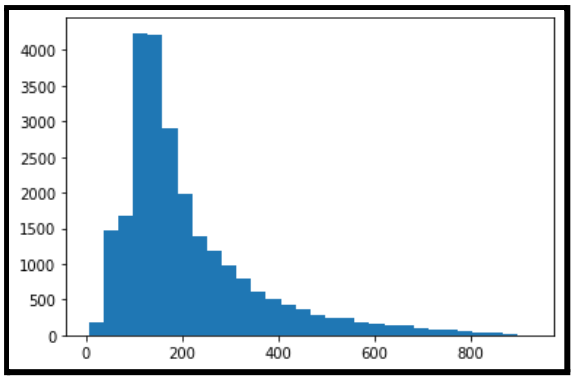

Imagem 1: Histograma dos dados ***tokenizados***

Para que os modelos pudessem passar para as próximas etapas do processo, foram realizados tratamentos nos *tokens* de ***teste***, ***treinamento*** e ***validação*** onde através da função *pad_sequences*, os dados ***tokenizados*** foram transformados em vetores com mesmo tamanho, *pad_sequences* é usado para garantir que todas as sequências em uma lista tenham o mesmo comprimento. Por padrão, isso é feito por preenchimento de 0 no início de cada sequência até que cada sequência tenha o mesmo comprimento da sequência mais longa 850. Finalizando o preparo dos dados antes de introduzi-los ao treinamento, os dados dos rótulos de ***teste***, ***treinamento*** e ***validação*** do *dataset* ***imdb pt-br*** foram transformados em vetores *binárias*, ou seja, vetores contendo *0* e *1*, utilizando o one-hot-encoding que significa transformá-las em variáveis(colunas) e binárias; Ao finalizar o preparo dos dados, os mesmos foram introduzidos ao modelo ***Keras*** (***rede neural recorrente LSTM***) que por sua vez teve seu tamanho da entrada de dados definida com os 850 ***tokens*** verificados no *histograma* apresentado na *Imagem 3*, junto da transformação dos tokens em vetores densos, com uma definição de 20.000 palavras de tamanho de vocabulário e 300 de saída, também definindo a quantidade ideal de neurônios que a rede ***LSTM*** irá processar em sua parte temporal definidas em 256 *features*, com retorno de ativação dos 850 tokens verificados, fazendo o pooling, que nada mais é que a quantidade de instâncias por 256 *features* pegando o valor máximo para cada *dimensão* e por fim definindo a configuração da classificação onde foi definido a saída de N x 2 *neurônios* (um para ***positivo*** e um para ***negativo***), ativando a *softmax* que converte o resultado de saída em resultados *probabilísticos*.

Para executar treinamento da ***rede neural recorrente LSTM***, foram definidas o uso de 30 *épocas* com uma taxa de aprendizado de 0.04 e tamanho de 512 *batch*, tendo seu treinamento otimizado utilizando o modelo de otimização ***Adam*** (***é um método de SGD que usa uma estimativa adaptativa dos momentos de primeira ordem e momentos de segunda ordem.***), com a função de custo *Categorical Cross-entropy* (Entropia cruzada categórica) e a métrica de resultado definida pela acurácia; Após o treinamento do modelo, pode-se observar o histórico de perdas que o modelo gerou, conforme observa-se na *Imagem 2*.

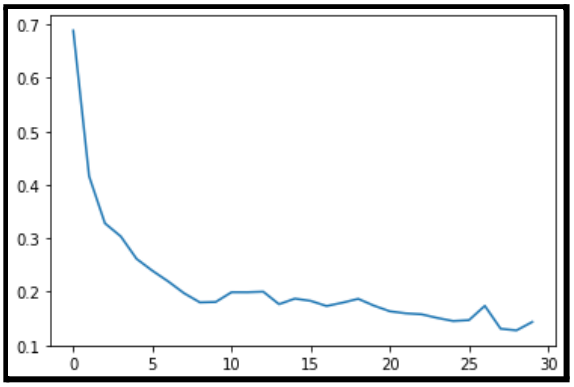

Imagem 2: Gráfico de perdas do treinamento da ***rede neural recorrente LSTM.***

Após o treinamento da ***rede neural recorrente LSTM***, os dados treinados foram salvos e carregados em um modelo para serem utilizados no teste de *predição*, onde através de uma classe de predição os dados treinados, junto dos textos que foram ***tokenizados*** passaram por uma predição de avaliação de sentimentos sob um conjunto de textos aleatórios, a fim de validar a assertividade da classificação; Ao realizar o teste e sua ***validação***, foram realizadas as avaliações dos modelos, começando pelo modelo testado com os dados ***tokenizados*** e os dados do dataset imb pt-br tratado, tendo seu resultado de teste demonstrado baixo.

Métricas de avaliação: precision: 0.4788 recall: 0.4867 fscore: 0.4827

Acurácia: 0.8088

Ao finalizar o treinamento e a predição dos dados do dataset ***imdb pt-br***, tanto no modelo ***multinomialNB***, quanto na ***rede neural recorrente***

LSTM, com os modelos treinados o *dataset* com todos os ***NPS*** da *Safeweb* foram passados para serem *classificados* em ambos modelos, assim apresentando a *classificação* do sentimento ***positivo*** e ***negativo*** de cada avaliação de ***NPS***, resultando em um novo *dataset* acrescentando duas novas colunas contendo os resultados das avaliações ***positivas*** e ***negativas*** do modelo ***MultinomialNB*** e da ***rede neural recorrente LSTM***, resultando em um *dataset* com 10.325 registros por 9 colunas; em seguida foram convertidas todas as *notas* registradas no *dataset* para valores inteiros, para que fosse possível cruzar as avaliações *quantitativas* com as avaliações *qualitativas*. Podemos observar o resultado da classificação sob o *dataset* de ***NPS***, conforme a *Imagem 3*, sendo esse resultado obtido pelos modelos ***MultinomialNB "Scikit Naive Bayes"*** e ***rede neural recorrente LSTM,*** mostrando uma tabela com os contadores de comentários que foram classificados como ***positivos*** e ***negativos*** para cada subgrupo do *NPS* "***Promotores (9 e 10)***", "***Passivos (7 e 8)***" e ***"Detratores (0 a 6)***".

Promotores (9 e 10)	MultinomialNB	Keras	MultinomialNB/Keras
Positivo	7196	7294	7196
Negativo	2681	2583	1262
Total	9877	Diferença	2740
Passivos (7 e 8)	MultinomialNB	Keras	MultinomialNB/Keras
Positivo	101	128	77
Negativo	96	69	96
Total	197	Diferença	75
Detratores (0 a 6)	MultinomialNB	Keras	MultinomialNB/Keras
Positivo	51	99	34
Negativo	200	152	135
Total	251	Diferença	82
Total Registros	10325		

Imagem 3: resultado das classificações.

RESULTADOS E CONSIDERAÇÕES FINAIS

De acordo com a realização do trabalho, pode-se observar a diferença entre as acurácias disponibilizadas pelos modelos ***MultinomialNB*** e ***rede neural recorrente LSTM***, onde para o modelo ***MultinomialNB*** obteve-se a *assertividade* da *acurácia* resultando 0.9143, enquanto para a ***rede neural LSTM*** obteve-se a *assertividade* de 0.8088. Após a análise das *acurácias* apresentas, percebeu-se que o modelo de *classificação* mais eficiente para ser utilizado para este tipo de trabalho foi o modelo ***MultinomialNB***, já que sua pontuação ***fscore*** (*média harmônica ponderada da precisão e recordação, onde uma pontuação fscore atinge seu* ***melhor*** *valor em 1 e 0 para o* ***pior*** *valor.*) obteve sua precisão em 0.9123, enquanto para o modelo de ***rede neural recorrente LSTM*** obteve sua pontuação ***fscore*** em 0.4827, concluindo então que o modelo ***MultinomialNB*** seria o mais indicado, já que obteve a melhor *assertividade* em seus resultados durante a *classificação* dos dados de ***NPS***.

REFERÊNCIAS

Joel Grus"Data Science do Zero - Primeiras Regras com o Python", Alta Books, 2016

Tatiana Escovedo, Adriano Koshiyama "Introdução à Data Science - Algoritmos de machine Learning e métodos de análise", Casa do Código, 2020

Guilherme Silveira, Bennett Bullock"Machine Learning - Introdução a classificação",Casa do Código, 2018

Eduardo Corrêa,"Pandas Python - Data Wrangling para Ciência de Dados",Casa do Código , 2019

LuísFred,"IMDB-PT-BR", <https://www.kaggle.com/luisfredgs/imdb-ptbr>

Ricardo Silveira Martins de Mello, "Código fonte", <https://github.com/ricardosmdemello/TCC_PUCRS_ANALISE_DE_SENTIMENTOS_NPS>, 2021