A ESTATÍSTICA POR TRÁS DA INFLUÊNCIA

Arthur Machado Pires de Camargo; Guilherme Moreira Rodrigues; Tamara Nubia Nunes Matamala; Victor Miguel de Oliveira; Wallace Machado Moreira Pontes

Orientador: Rodrigo Bossini

Resumo: Neste artigo vamos analisar a influência dos tweets no mercado financeiro, comparando tweets de pessoas comuns com uma pessoa influente, treinando redes neurais com diferentes métodos para prever se a variação do valor de abertura e fechamento da bolsa de valores foi positiva ou negativa.

Palavras-chave: Estatística, Redes Neurais, Mercado Financeiro, Twitter, Elon Musk

THE ESTATHISTIC BEHIND THE INFLUENCE

Abstract: In this article we will analyze the influence of tweets on the financial market, comparing tweets from ordinary with an influential person, training neural networks with different methods to predict if the variation of the opening and closing value of the stocks will be positive or negative.

Keywords: Estathistic, Neural Network, Financial Market, Twitter, Elon Musk

1. Introdução

A tecnologia com o passar do tempo vem ajudando de diversas formas investidores no processo de avaliação de investimentos financeiros, identificando mudanças e indicando o melhor momento para o investimento usando as mais refinadas técnicas da a área da Inteligência Artificial (IA), como por exemplo o aprendizado de máquina, que busca entender as diversas variações do mercado financeiro a partir de uma base de dados histórica.

Com o aperfeiçoamento da IA ao longo dos anos, a tecnologia passou a ser utilizada em diversos processos inteligentes que auxiliam no reconhecimento de padrões, na tomada de decisão ou na automação de tarefas repetitivas.

Essas técnicas são utilizadas para descoberta de padrões através de classificações de dados, que na maioria das vezes seria impossível para um ser humano classificar em tão pouco tempo.

Essa área da computação concentra-se em resolver problemas, processando grandes quantidades de dados e obtendo resultados antes inimagináveis. A IA tem um papel importante no processo de análise e descoberta de informações, não só no mercado financeiro, mas também ajudando a sociedade na área da saúde, por exemplo. Com a IA consumindo as informações relevantes através dos dados passados é possível prever fatos e apontar possíveis melhorias para se obter um resultado mais certeiro.

Usando a IA para prever possíveis resultados da bolsa de valores, a área em destaque da inteligência artificial, utilizada neste trabalho, são as chamadas técnicas de aprendizado de máquina.

A técnica de aprendizado de máquina (AM) é uma ramificação da Inteligência Artificial, que vinha sendo almejada a alguns anos, nasceu da curiosidade de pesquisadores da área de que seria possível as máquinas aprenderem a partir de reconhecimento de padrões de dados, porque até então as máquinas só faziam atividades das quais elas eram instruzıdas constantemente. Além de resolver problemas de banco de dados, é capaz de se adaptar

ao ambiente, pois o sistema aprende com processamentos anteriores e se adapta livremente quando expostos a novos dados, capazes de produzir resultados confiáveis e sem repetições, Sousa, Maria Cristina Cordeiro Sousa (2020). Logo, a IA ajudaria a apoiar decisões que não seriam tomadas sem os resultados destacados pela execução dos algoritmos de aprendizagem.

Neste artigo iremos utilizar algoritmos de machine learning para realizar um estudo da possibilidade de prever a variação do mercado financeiro com auxílio do Twitter. Utilizando como base para estudo, bases de tweets de pessoas comuns em comparação com uma pessoa influente (Elon Musk), comentando sobre o mercado financeiro e analisando a correlação dos tweets com uma base da bolsa de valores onde temos a variação da bolsa nos dias em que os tweets foram realizados.

Dada a relação de CEO com a Tesla e a fama do Elon Musk esse estudo também será interessante para orientar uma discussão sobre o poder da influência nos dias de hoje, e que dependendo do resultado podemos concluir que a influência é capaz de impactar até mesmo o mercado financeiro.

1.1 Justificativa

Hoje a grande maioria das pessoas utilizam as redes sociais como referência para tendências de moda, jogos, estudos, receitas e etc. Com as redes sociais fazendo cada vez mais parte das nossas vidas, acreditamos que esse estudo seja de grande valor, uma vez que as redes já possuem grande influência no mercado de consumo. Uma irrefutável prova disso são os grandes valores pagos para que influenciadores com um número expressivo de seguidores e bom engajamento realizem publicidades a cada post realizado em seus perfis pessoais. Com esse estudo será possível verificar a possibilidade de prever a variação de ações do mercado financeiro com base em tweets de uma pessoa influente. Uma vez que esse estudo retornar resultados positivos, acreditamos que seria interessante desenvolver inteligências artificiais que realizem análises de tweets em tempo real para auxiliar na tomada de decisão de compra e venda de ações. Seria mais uma ferramenta para auxiliar na previsão do mercado financeiro, um dos maiores sonhos do homem do século XXI.

1.2 Objetivos (Geral e específicos)

O objetivo deste artigo é realizar um estudo de mercado financeiro, verificando a possibilidade de estimar se a variação entre o valor de abertura e fechamento da bolsa de valores é positivo ou negativo utilizando os posts da rede social chamada Twitter como base dessa previsão. Para isso, vamos utilizar o VADER (Valence Aware Dictionary and sEntiment Reasoner), um algoritmo otimizado para textos de redes sociais que funciona por meio de dicionário léxico para análise de sentimentos, que classifica palavras e frases como positivas, neutras ou negativas. Para que possamos ter um viés de confirmação a respeito dessa polaridade dos textos, também utilizamos o AFFIN, outro algoritmo que realiza a análise de sentimentos a partir de textos, separando-os em textos positivos, neutros ou negativos. Outro fator que levamos em consideração para enriquecer a análise e adicionar uma avaliação extra no impacto dos tweets, foi verificar se o engajamento de uma pessoa afetaria na sua capacidade de influenciar o mercado financeiro, comparando os resultados de uma base de tweets de pessoas comuns sobre o mercado de ações relacionadas à empresa Tesla com uma base de tweets de uma pessoa influente e intimamente relacionada a empresa. Por fim, analisando a correlação dos tweets com a bolsa de valores contendo a variação da mesma nas datas correspondentes aos tweets em questão.

2. Revisão Bibliográfica

Durante pesquisas encontramos alguns estudos interessantes, dentre eles encontramos o “Predicting Stocks Movement using Social Media Analytics.”(FOTIOS, 2018) que destaca muito bem como o mercado de ações pode ser previsto, com uma eficácia satisfatória, efetuando uma análise emocional de mídias sociais, inclusive destaca como o Twiter em específico se demonstra uma ótima rede social para realizar esse tipo de análise por seu foco em posts textuais e curtos combinados a sua comunidade extremamente ativa e, o mais importante, a api pública disponibilizada para desenvolvedores que é ótima para construir uma base de dados completa com posts de múltiplos usuários. Porém diferente do artigo mencionado anteriormente, nosso foco seria tentar definir como a influência e o poder de uma única pessoa poderia impactar o mercado financeiro.

Em outras palavras nosso teste pretende estabelecer um paralelo entre o Elon musk e as ações da Tesla assim como o paralelo feito entre o ex-presidente americano Donald Trump e o mercado de ações internacional, relação estudada no “O Impacto nos Mercados Financeiros dos Tweets do Presidente Norte Americano Donald Trump”(MALAQUIAS FILHO, 2021). A ligação entre o mercado financeiro de uma ou mais empresas e uma única pessoa em ambos os casos se exalta pela fama da pessoa somada a um cargo de poder extremamente relevante para as empresas observadas.

Em nosso projeto o Elon Musk revelou ser um material de estudo excelente, por ter uma relevância e engajamento mundial nas redes sociais, além de ser algo bem similar ao que pode ser definido como embaixador para a Tesla, em resumo um representante da marca no ponto de vista de marketing, sendo visível uma relação íntima entre a imagem pública da empresa e a imagem pública da pessoa, como pode ser visto no “The Influence of Brand Ambassador on Brand Image and Consumer Purchasing Decision: a Case Of Tous Les Joursin Indonesia.”(WANG; HARIANDJA, 2016). Claro que além dessa relação entre marca e embaixador o Elon Musk também possui um cargo de poder relevante para a Tesla sendo o CEO da mesma.

3. Metodologia

O primeiro passo para o projeto foi localizar ou criar uma base de dados para que possamos estudar, sabendo disso em pesquisas conseguimos localizar no Kaggle (site com múltiplos datasets, códigos e competições voltadas para data science) a publicação “How Twitter affects the Stock Market” (ALI,2021) contento os dados de tweets de pessoas aleatórias falando sobre as 10 empresas no topo da Nasdaq e as ações das mesmas empresas, no período de 2010 a 2020. Com isso associamos os tweets com as ações das empresas a partir de suas datas, filtramos os dados para que ficassem somente os tweets e as ações referentes a Tesla, assim conseguimos nossos dados de controle do experimento contento 1.096.868 tweets.

Agora para os dados relacionando os tweets especificamente do Elon Musk com as ações da Tesla passamos por outro processo. Ainda no Kaggle conseguimos localizar o post “Elon Musk Tweets (2010 - 2022)” (DALILA,2022), que continha um registro bem completo dos tweets feitos pelo Elon Musk, então baixamos os registros desse post e fomos diretamente no site da Nasdaq para conseguir a um registro de ações da Tesla completo para que possamos fazer a relação pelas datas, dessa forma conseguimos os dados do nosso alvo de estudos contendo 10.792 tweets que é uma base relativamente pequena porém suficiente para o estudo.

Definidos os dados a serem estudados começamos a estipular qual seria a melhor linguagem e as melhores tecnologias para prosseguir com o projeto e depois de algumas

discussões concluímos que a melhor linguagem para o estudo seria o Python pelas bibliotecas exclusivas voltadas para data science que a mesma possui.

3.1 Estruturando os dados

Com os dados tabulares vindo de diferentes fontes, precisamos a princípio relaciona-los e organizá-los para que sigam um padrão, e se possível angariar algumas informações sobre esses dados para uma análise descritiva e essa funções foram cumpridas principalmente desfrutando das bibliotecas Pandas e Numpy, essas bibliotecas se destacam bastante no mercado de data science e e foram peças chaves para o projeto então seguiremos com uma breve descrição da funcionalidade de cada uma.

3.1.1 Pandas

O Pandas é uma biblioteca do python que facilita a visualização, manipulação e os cálculos de dados tabulares (tabelas), que matematicamente falando podem ser rusticamente comparados a matrizes. Usando ela modelamos os dados para que fiquem dispostos em uma linha diretamente uma relação entre os textos dos tweets, os indicadores de engajamento da rede social e os dados da bolsa relacionados ao dia de publicação do tweet. Além disso a biblioteca teve importância para a análise descritiva feita na futuramente no projeto e é importante destacar alguns métodos da mesma que possibilitaram isso

- Head é um método para listar as x primeiras linhas de um data frame;
- Info imprime informações sobre um DataFrame, incluindo o dtype e as colunas do índice, valores não nulos e uso de memória;
- Describe calcula um resumo das estatísticas referentes às colunas (variáveis) do DataFrame;
- Values Count retorna o objeto contendo contagens de valores únicos. O objeto resultante estará em ordem decrescente para que o primeiro elemento seja o elemento que ocorre com mais frequência;
- Corr encontra a correlação em pares de todas as colunas no dataframe.

3.1.2 Numpy

Numpy(Numerical Python) é uma biblioteca mais voltada a facilitar a manipulação e processamento de matrizes multidimensionais e foi mais eficaz na análise descritiva anteriormente mencionada, valendo a pena destacar os seguintes métodos.

- Shape: retorna uma tupla com cada índice com o número de elementos correspondentes em entre colunas e linhas.
- Corrcoef: retorna os coeficientes de correlação produto-momento de Pearson. Fazendo uma relação entre a matriz do coeficiente de correlação, R , e a matriz de covariância, C.

3.1.3 Preparação dos Dados

Depois de encontrar as bases que servirão como fonte de estudo, é necessário preparar as mesmas para que seja possível relacioná-las. A base de Tweets de pessoas comuns falando sobre o mercado financeiro, “How Twitter affects the Stock Market” (ALI,2021), precisou passar por uma série de tratamentos e filtros antes de ser analisada. O primeiro passo foi filtrar a coluna que contém os nomes das empresas relacionadas aos tweets “ticker_symbol”, apenas com a empresa Tesla (“TSLA”), depois renomeamos as colunas que não tinham nomes intuitivos, para facilitar a nossa análise. Por fim, foi necessário realizar criar uma lógica para relacionar os tweets feitos após as 17h de todo último dia útil da semana, com o próximo dia útil, pois uma vez que esse é o horário de fechamento da

bolsa de valores, entendemos que tweets realizados após esse horário irão afetar o seu próximo momento de abertura como indicado na figura a seguir(figura 1).

Figura 1 – Código para relacionamento correto entre tweets e ações

```
[ ]
#relacionando os tweets do elon musk com as ações da tesla pela data
#ja ajustando os tweets para que caso sejam feitos no periodo que a bolsa esteja fechada, sejam relacionados ao proximo momento que ela abrir
from re import X
from datetime import datetime, timedelta
import numpy as np
stock_days = stocks_tesla['New_Date'].tolist()
elon_days = elon_tweets['date'].tolist()
disorder = True
cicles = 1
output = []

print('tamanho da lista inicial->', len(elon_days))
while disorder == True:
  mistake_list = []
  output = []
  print('resetou ->',len(output))
  for x, item in enumerate(elon_days):
    if item not in stock_days:
      mistake_list.append(item)
      if cicles <= 7:
        example = datetime.strptime(item, '%Y-%m-%d') + timedelta(days=1)
        elon_days[x] =  example.strftime("%Y-%m-%d")
    else:
      output.append(item)

  if len(mistake_list) == 0 or cicles >= 7:
    print("Todos os itens tem correspondência")
    print('tamanho da lista final->', len(output))
    disorder = False
  else :
    print(cicles, '- item sem correspondência ->', len(np.unique(np.array(mistake_list))), 'itens restantes->', len(output))
    cicles = cicles + 1
    disorder = True
tesla_days = output
```

Fonte: Própria (2022)

Já a base de tweets do Elon Musk, “Elon Musk Tweets (2010 - 2022)” (DALILA,2022), precisou ter seus anos filtrados, uma vez que a mesma possuía registros desde 2010, já que base de pessoas comuns possuía tweets a partir de 2015. A base do Elon Musk, veio separada em 12 bases diferentes, uma para cada ano, então empilhamos as bases que precisaríamos utilizar. Foi necessário também realizar o mesmo tratamento de relacionar os tweets realizados após as 17h do último dia de cada semana com o próximo dia útil.

3.2 Processamento dos textos

Com os dados agrupados e organizados em um padrão, precisamos seguir para etapa seguinte que seria processar de alguma maneira os textos dos tweets em valores numéricos que representem algo relevante, para isso usamos as bibliotecas chamadas Afinn e Vader, que conseguem definir se um texto é positivo, negativo ou neutro. Mas antes de podermos aproveitar desse processamento, seria necessário normalizar o texto, deixando, tudo em minúsculo, retirando links, retirando valores numéricos dentre outras ações que farão mais sentido quando for explicado o funcionamento do Afinn e do Vader. Para fazer essa normalização encontramos ainda no kaggle a postagem “K417 - How Twitter affects the Stock Market” (YAMAMOTOYUDAI,2021) que mostra um exemplo de aplicação do Affin e do Vader com os dados das 10 empresas no topo da Nasdaq mencionando anteriormente, nele podemos encontrar um trecho de código que realiza a normalização do texto para o uso das bibliotecas, então o utilizamos.

Figura 2 – Trecho de código do projeto originado no Kaggle

```
# algoritimo para a normalização dos textos
import re
def remove_special_character(tweet):
    # remove the old style retweet text "RT"
    tweet = re.sub(r'^RT[\s]+', '', tweet)

    # remove hyperlinks
    tweet = re.sub(r'https?:\/\/.*[\r\n]*', '', tweet)

    # remove hashtags. only removing the hash # sign from the word
    tweet = re.sub(r'#', '', tweet)

    # remove single numeric terms in the tweet.
    tweet = re.sub(r'[0-9]', '', tweet)

    return tweet
```

Fonte: Própria (2022)

Feita a normalização passamos para a etapa de processar de fato os textos, para que fique claro a teoria por trás desse processamento explicaremos a seguir o funcionamento do Afinn e do Vader.

3.2.1 AFINN

AFINN se trata de um algoritmo que tenta definir uma polaridade, negativa, positiva ou neutra para um texto utilizando um dicionário lexical, que atribui uma intensidade negativa ou positiva para cada vocábulo de uma linguagem, possibilitando o cálculo de uma nota que representa a polaridade média de um texto.

3.2.6 VADER

VADER(Valence Aware Dictionary and sEntiment Reasoner) assim como o AFINN se trata de um algoritmo que tenta atribuir uma polaridade a um texto utilizando um dicionário lexical calculando polaridade média de um texto. Porem o mesmo tem um diferencial, pois o dicionário lexical dele é especialmente ajustado para um desempenho mais eficiente com textos presentes em redes sociais.

3.3 Análise descritiva

Feito o processamento do texto conseguimos um valor para cada texto representando o quão negativo ou positivo o mesmo era, e já podemos relacionar isso às ações da tesla, porém percebemos que ainda existia um ajuste que deveria ser feito antes disso.

Os tweets estavam relacionados de um para um com os registros de um dia inteiro das ações da Tesla, e essa relação não representava bem a realidade, pois mais de uma pessoa poderia falar sobre a Tesla no mesmo dia estabelecendo uma relação de um para muitos. Resolvemos isso agrupando os tweets pelas datas fazendo uma média das polaridades atingidas com o Vader e o Afinn, e a média dos medidores de engajamento do twitter (like, comentário e retweet), deixando os dados dispostos como pode ser visto na figura 3

Figura 3 – Disposição final dos dados do projeto

Tabela com descrição de cada variável

Variavel	Descrição	Tipo
Like	Número de likes no tweet	Quantitativa
Comment	Número de comentários no tweet	Quantitativa
Rt	Número de retweets no tweet	Quantitativa
Volume	Quantidade de ações negociadas durante o dia do tweet	Quantitativa
Open	O preço de abertura da ação na bolsa	Quantitativa
Close	O preço de fechamento da ação na bolsa	Quantitativa
Variation	O tanto que a bolsa variou no dia, do momento que abriu até o momento que fechou	Quantitativa
Engagement	Soma dos valores normalizados de Like, Comment e RT	Quantitativa
Afinn	Algoritmo que atribui nota para a positividade ou negatividade do texto, quanto mais alta a nota mais positivo	Quantitativa
Vader	Algoritmo que atribui nota para a positividade ou negatividade do texto, quanto mais alta a nota mais positivo	Quantitativa

Fonte: Própria (2022)

Agora com uma relação correta estabelecida entre os tweets e as ações da Tesla decidimos fazer uma análise descritiva para descobrir se seria possível estabelecer alguma relação simples de influência dos tweets nas ações. Para executar essa análise descritiva, além das bibliotecas anteriormente citadas, também utilizamos o Matplotlib e o Seaborn e detalharemos em seguida como cada uma dessas bibliotecas contribuíram.

3.3.1 Matplotlib

Matplotlib se trata de uma biblioteca que possibilita a criação e estilização de gráficos e figuras, ele foi de extrema importância para a exibição de alguns gráficos simples do projeto, tendo em destaque os seguintes métodos.

- Hist, método usado para gerar histogramas;
- Subplots, Adjust método utilizado para se plotar gráficos em um grid;
- Add Subplot, método utilizado para organizar gráficos em formato matricial de x dimensões.

3.3.2 Seaborn

Seaborn é uma biblioteca de visualização de dados baseada na biblioteca matplotlib. Ela fornece uma interface de alto nível para desenhar gráficos estatísticos, basicamente expandindo as possibilidades da biblioteca anterior, e mantendo o nível de facilidade para gerar gráficos mais complexos. Os métodos mais importantes do Seaborn que usamos foram as seguintes.

- Boxplot, mostra os quartis do conjunto de dados enquanto os bigodes se estendem para mostrar o resto da distribuição dos dados, apresentando também os outliers;
- Heatmap, método que retorna o nível de eixos e desenha o mapa de calor nos eixos atualmente ativos.

3.4 Treinando modelos inteligentes

Os resultados da análise descritiva não apresentou resultados muito relevantes da influência dos tweets nas ações da Tesla, pelo contrário demonstrou que há poucos índices que possam comprovar matematicamente a influência dos tweets. Porém ainda há outra

forma de testarmos essa relação, que no caso seria treinar uma IA para tentar fazer essa relação. Esse feito exigiu em nosso projeto a utilização de uma biblioteca bem conhecida no meio da inteligência artificial, o Scikit Learn.

Scikit Learn é uma biblioteca que facilita o acesso a técnicas de machine learning com a linguagem python, seja para classificação ou casos preditivos, a biblioteca existe desde 2007 e é código aberto.

Essa biblioteca é repleta de diferentes modelos de IA e alguns métodos para otimização do funcionamento desses modelos, como ela facilita muito a aplicação de modelos complexos decidimos treinar vários modelos e levar em consideração sempre o que apresentar o melhor desempenho. Então seguiremos com uma explicação sucinta da teoria por trás de cada um dos modelos presentes no projeto.

- GaussianNB (Gaussian Naive Bayes) é um modelo de IA baseado no algoritmo Naive Bayes, que tenta determinar uma probabilidade a posteriori multiplicando a probabilidade a priori pela probabilidade de cada classificação e normalizando-as para que somadas possam equivaler ao todo;
- DecisionTreeClassifier, esse algoritmo se resume a definir pontos de decisão chamados “nós” que usam um valor como fator decisivo entre 2 possibilidades de classificação. Em um exemplo mais sólido, podemos assumir que um nó diria se você precisa se alimentar utilizando sua fome como fator de decisão. Juntando vários desses Nós o algoritmo da árvore de decisão consegue fazer classificações de altas complexidades atrelando os nós às colunas presentes nos dados de entrada do modelo;
- RandomForestClassifier esse modelo simplesmente gera várias árvores de decisão treinadas com amostras aleatórias dos dados e determina uma classificação pela média dos resultados dessas árvores criadas;
- LogisticRegression é um algoritmo usado exclusivamente para classificações binárias, ele utiliza da equação da curva logística para determinar a proximidade de um item da parte inferior ou superior da curva podendo assim classificá-lo em uma das possibilidades;
- MLPClassifier(Multi Layer Perceptron Classifier) é um modelo classificador que tem como base uma rede neural com uma ou mais camadas escondidas com n neurônios, esses “neurônios” mencionados são o conjunto de partes que se conecta para compor a rede neural e sua concepção é em resumo um modelo matemático que produz um número positivo ou negativo dado alguns inputs e pesos para os mesmos. Esse modelo matemático foi criado em 1940 por Warren McCulloch e Walter Pitts respectivamente um neurofisiologista e um matemático;
- SVC (Suport Vector Classifier) é um modelo de classificação que tenta encontrar um hiperplano em um espaço N dimensional que consiga fazer a melhor separação dos dados entregues, podendo classificar dados posteriormente simplesmente determinando a posição do novo fator em relação a esse hiperplano.

3.5 Otimizando os modelos

Para ter certeza de que estávamos conseguindo o melhor resultado possível dos modelos selecionados, fizemos questão de implementar duas técnicas para otimizar os resultados, sendo elas a validação cruzada e o “grid search”.

Ao treinar um modelo de inteligência artificial você sempre precisa fazer uma separação entre as partes dos dados que irão treinar o seu modelo e a parte dos dados que irá servir para calcular a eficiência do modelo de maneira empírica, esses grupos são chamados respectivamente de grupo de treino e grupo de teste. A validação cruzada se baseia em separar o seu dado em x partes e verificar o desempenho do modelo o mesmo número x

de vezes, porém selecionando uma parte diferente dessas divisões para ser considerado o grupo de teste a cada iteração, enquanto todas as outras partes serão consideradas parte do grupo treino. isso em teoria possibilita encontrar a melhor combinação disponível entre grupos de treino e teste na sua base de dados.

A biblioteca Scikit Learn tem um método chamado K Fold que permite fazer uma validação cruzada de forma manual, ela basicamente faz a separação e o loop necessário para que você realize a validação cruzada como desejar,e foi esse o método que usamos para implementar essa técnica.

É importante destacar que apesar dos modelos de IA citados anteriormente terem o mesmo objetivo de serem treinados para tentar predizer se as ações da tesla irão subir ou descer com base nos tweets postados no dia, cada modelo tem sua própria lógica com diferentes parâmetros que podem ser configurados de diferentes maneiras. Uma Decision Tree pode ter um número variados de nós uma RandomForest pode ter um número variado de árvores, um Multi Layer Perceptron pode um número x de camadas ocultas com x neurônios, um Suport Vector Classifier pode ter diferentes funções de ativações, dentre outros. Esses parâmetros variáveis de acordo com a lógica do modelo se chamam de hiperparâmetros, e a técnica de grid search consiste em definir um intervalo de possibilidades para esses hiperparâmetros e testar todas essas possibilidades para definir qual é a melhor ou quais são as melhores sabendo que os modelos geralmente têm mais de um hiperparâmetro.

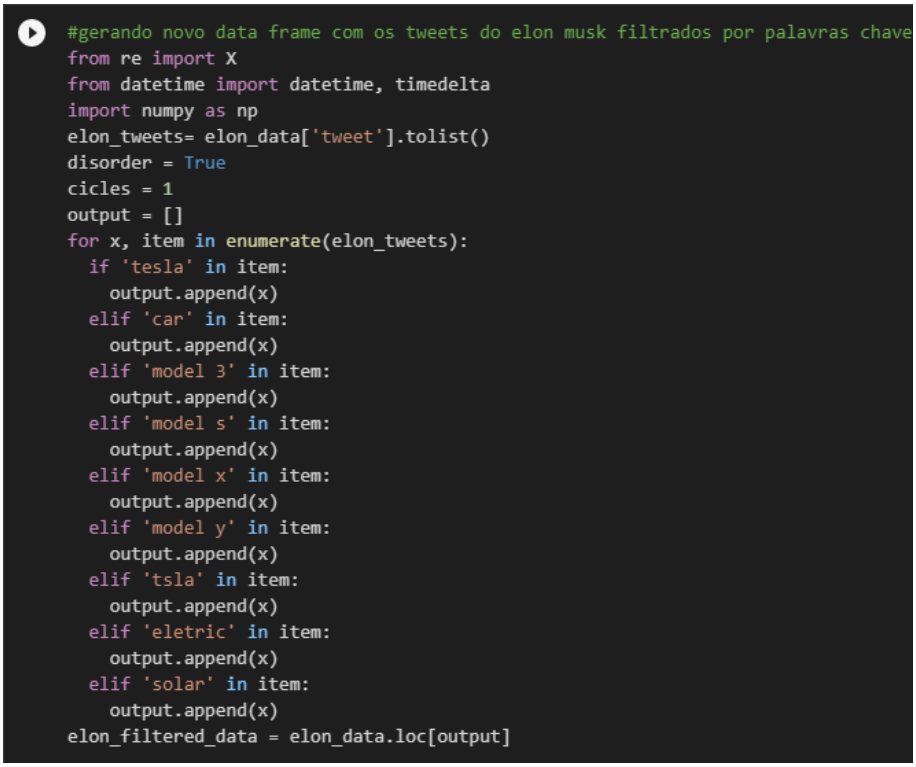
Com algumas listas de hiperparâmetros e alguns loops unidos ao K Fold conseguimos criar um código que aplica as 2 técnicas, reportando em tempo real a melhor acurácia atingida até o momento e a acurácia atingida a cada iteração e ao final exibe um relatório de desempenho de cada modelo indicando a melhor configuração de cada um deles e claro o qual foi a configuração e modelo que apresentaram o melhor resultado.

3.6 Reinterpretando os dados

Com todos esses processos implementados tivemos enfim os resultados das inteligências artificiais treinadas com os dados exclusivos do Elon Musk e com os dados de controle contendo tweets de pessoas aleatórias falando sobre a Tesla. Impressionantemente os dados de controle surtiram um melhor resultado com as inteligências artificiais chegando a 62,84% de acurácia 1,14% a mais que a acurácia atingida com os dados do Elon Musk. Apesar de ser uma diferença relativamente pequena seria o suficiente para podermos deduzir que a visão das pessoas sobre o'que o Elon Musk faz em suas redes sociais não tem um resultado tão impactante nas ações da empresa.

Porém repensando nossos dados conseguimos pensar em uma maneira de otimizar os resultados. Em resumo, apesar de termos o registro completo dos tweets feitos pelo Elon Musk no período estudado, não demos importância ao assunto dos tweets dele. Então demos alguns passos atrás e filtramos os tweets dele mantendo somente os tweets que continham uma palavra chave relacionada a empresa Tesla e seus produtos, utilizando o código da figura 4.

Figura 4 – Disposição final dos dados do projeto



```
#gerando novo data frame com os tweets do elon musk filtrados por palavras chave
from re import X
from datetime import datetime, timedelta
import numpy as np
elon_tweets= elon_data['tweet'].tolist()
disorder = True
cicles = 1
output = []
for x, item in enumerate(elon_tweets):
  if 'tesla' in item:
    output.append(x)
  elif 'car' in item:
    output.append(x)
  elif 'model 3' in item:
    output.append(x)
  elif 'model s' in item:
    output.append(x)
  elif 'model x' in item:
    output.append(x)
  elif 'model y' in item:
    output.append(x)
  elif 'tsla' in item:
    output.append(x)
  elif 'eletric' in item:
    output.append(x)
  elif 'solar' in item:
    output.append(x)
elon_filtered_data = elon_data.loc[output]
```

Fonte: Própria (2022)

Depois desse procedimento reduzimos drasticamente nossa base inicial de tweets do Elon Musk que já era pequena resultando em somente 2.683 tweets. Vale ressaltar que para treinar as inteligências artificiais tivemos que agrupar os tweets por dias e isso implica que todas as bases tiveram uma redução drástica antes dos treinamentos. Porém foi um agrupamento necessário para que a relação entre tweets e ações fizesse sentido. Enfim, repetimos o treinamento dos modelos de inteligência artificial com os mesmos métodos e técnicas utilizados, só que treinando os modelos com esse novo conjunto de dados gerados. E chegamos a um surpreendente resultado de 75% de acurácia, no modelo MLP.

4.Resultados e Discussão

Primeiramente analisando a distribuição dos dados dos dois datasets podemos perceber uma grande diferença no engajamento, como podemos observar na figura 5 e 6:

Figura 5 – Estatísticas dataset anônimos

	like	comment	rt	volume	open	close	variation	engagement	afinn	vader
count	1827.0	1827.0	1827.0	1827.0	1827.0	1827.0	1827.0	1827.0	1827.0	1827.0
mean	3.0	0.0	1.0	6571267.0	269.0	269.0	-0.0	5.0	0.0	0.0
std	4.0	0.0	1.0	4327624.0	56.0	56.0	6.0	4.0	0.0	0.0
min	0.0	0.0	0.0	710277.0	142.0	144.0	-28.0	0.0	-0.0	-1.0
25%	1.0	0.0	0.0	3895815.0	222.0	220.0	-4.0	2.0	0.0	0.0
50%	2.0	0.0	1.0	5417984.0	258.0	258.0	0.0	3.0	0.0	0.0
75%	6.0	1.0	1.0	7642613.0	316.0	316.0	3.0	8.0	0.0	1.0
max	22.0	2.0	7.0	33597290.0	435.0	431.0	36.0	28.0	0.0	2.0

Fonte: Própria (2022)

Figura 6 – Estatísticas dataset Elon Musk

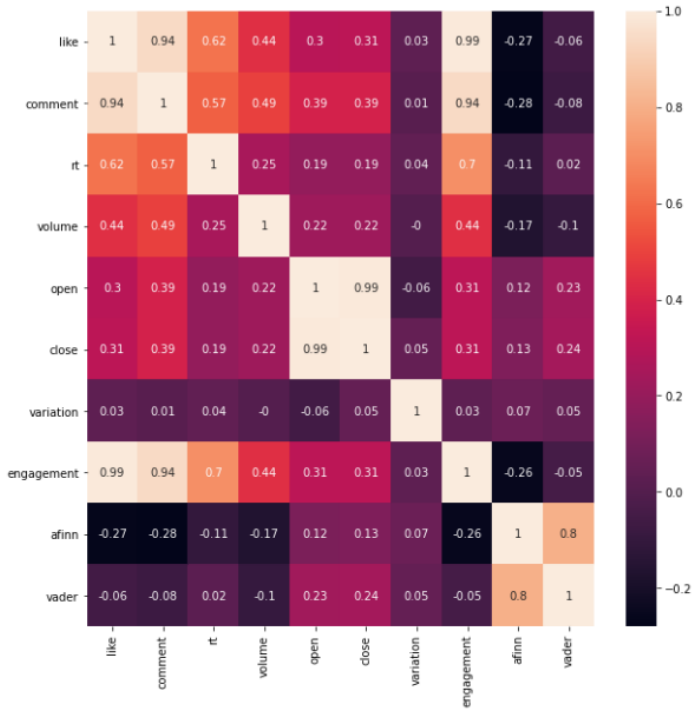
	like	comment	rt	volume	open	close	variation	engagement	afinn	vader
count	1413.0	1413.0	1413.0	1413.0	1413.0	1413.0	1413.0	1413.0	1413.0	1413.0
mean	21043.0	585.0	2638.0	46387419.0	111.0	111.0	0.0	24267.0	0.0	1.0
std	53522.0	1486.0	9643.0	31490761.0	128.0	128.0	2.0	63998.0	0.0	1.0
min	44.0	3.0	4.0	8285090.0	31.0	30.0	-9.0	59.0	-0.0	-4.0
25%	3249.0	153.0	339.0	25348010.0	49.0	49.0	-1.0	4000.0	0.0	0.0
50%	9220.0	295.0	921.0	36720380.0	62.0	62.0	0.0	10572.0	0.0	1.0
75%	22454.0	628.0	2210.0	58478800.0	76.0	76.0	1.0	25258.0	0.0	2.0
max	1596525.0	43536.0	302761.0	304693800.0	675.0	666.0	9.0	1942822.0	1.0	5.0

Fonte: Própria (2022)

A variável de engajamento foi criada a partir da soma dos comentários, likes e retweets e é fundamental para entendermos o alcance do tweet e seu impacto.

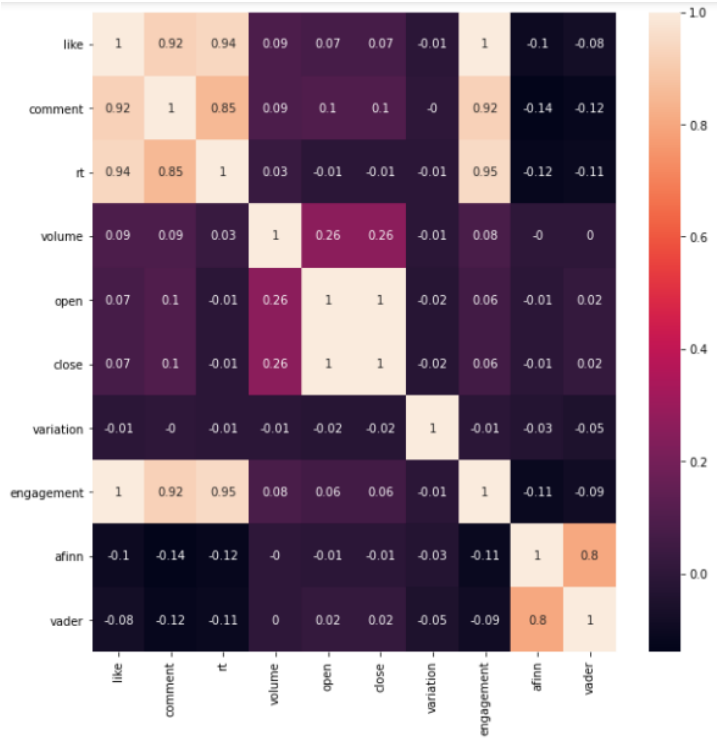
Abaixo nas figuras 7 e 8 podemos ver a correlação das variáveis sem nenhum tratamento no dataset, quanto mais escura, mais fraca é a correlação e quanto mais clara mais positiva a correlação:

Figura 7 – Correlação dataset anônimos



Fonte: Própria (2022)

Figura 8 – Correlação dataset Elon Musk

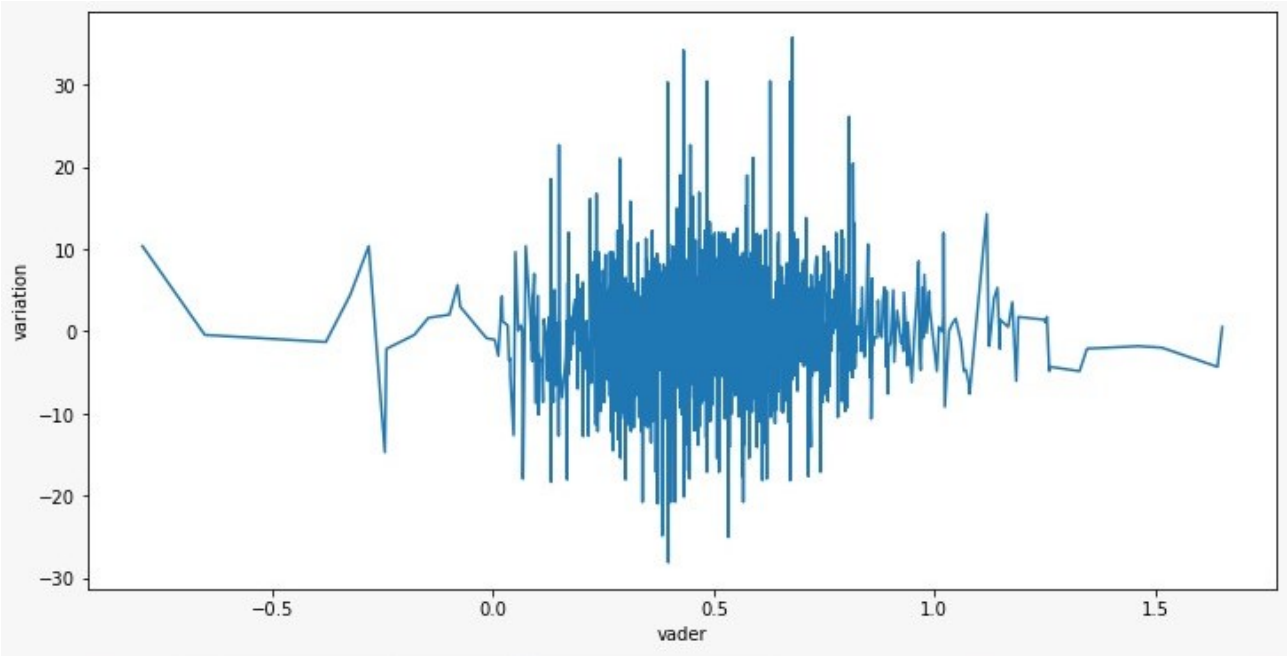


Fonte: Própria (2022)

Podemos perceber que a correlação dos tweets anônimos são mais fortes que as do Elon Musk.

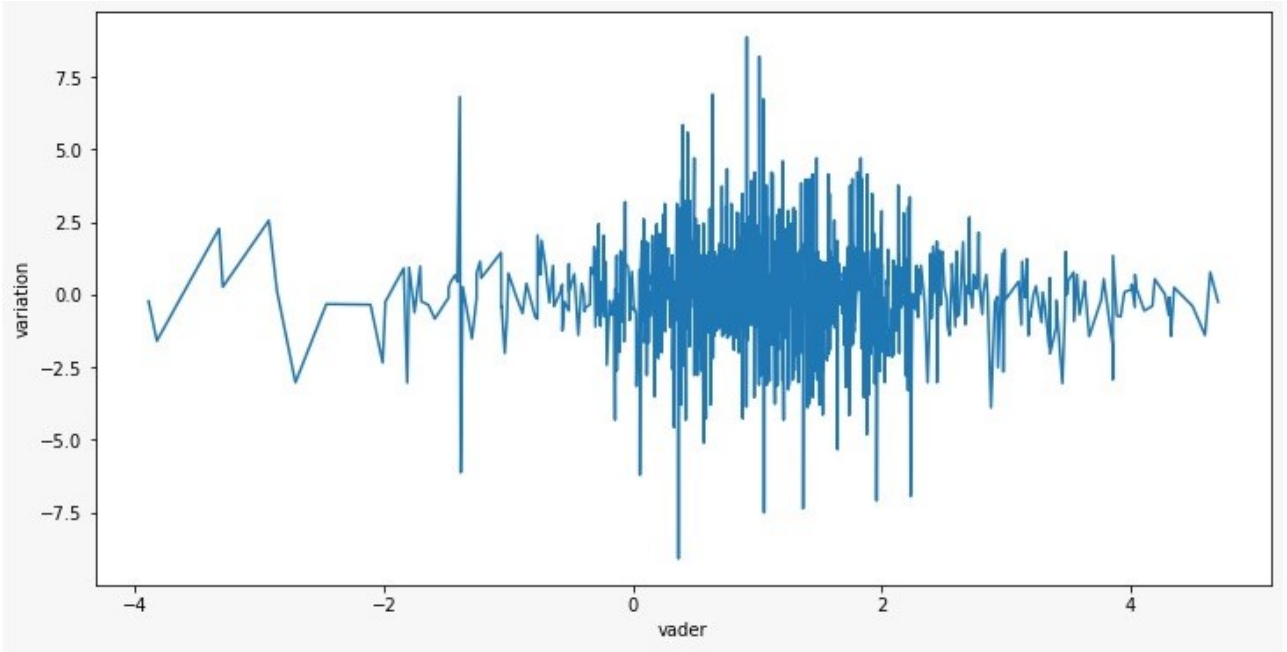
Com os dados obtidos acima e as devidas classificações dos tweets feita pelo Vader podemos enxergar melhor a relação tweet x variação da bolsa, nas imagens abaixo temos essa relação em forma de gráficos.

Figura 9 – Variação das ações x Tweets de pessoas anônimas



Fonte: Própria (2022)

Figura 10 – Variação das ações x Tweets do Elon Musk



Fonte: Própria (2022)

Com os resultados obtidos percebemos que quantos mais neutro o comentário mais complexo é predizer se a ação vai subir ou descer devido sua oscilação absurda, podemos notar isso nos gráficos acima, quanto mais próximo do zero (comentário neutro), mais imprevisível se torna a variação. Podemos observar também que esse comportamento não

ocorre só para os tweets de pessoas anônimas, mas também para os tweets do próprio Elon Musk.

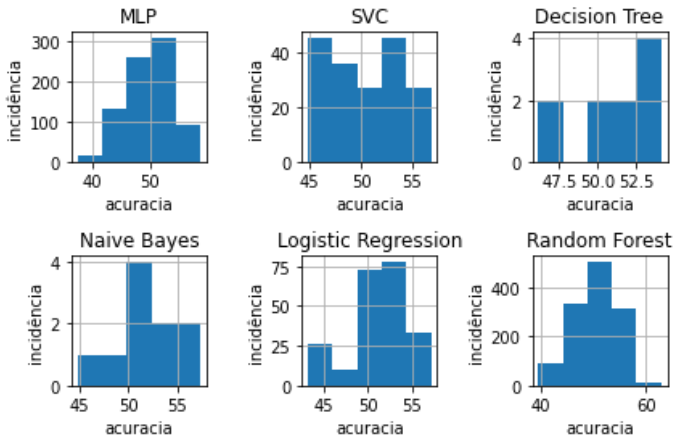
A princípio treinamos e testamos 6 modelos de IA para tentar predizer da maneira mais eficaz, utilizando métodos de grid search e cross validation para que se possa supor um valor médio e de desempenho dos modelos para o melhor desempenho possível, abaixo podemos ver os resultados de cada modelo nos dois datasets.

Figura 11 – Resultados dos treinamentos das IAs com os tweets de pessoas aleatórias

```
--------------------------------------------------------------------
MLP
A melhor nota:  58.791208791208796 %
melhor random_state:  1
melhor hidden_layer_sizes:  128
melhor conjunto KFold: 7
--------------------------------------------------------------------
SVC
A melhor nota:  56.830601092896174 %
melhor random_state:  0
melhor kernel:  sigmoid
melhor conjunto KFold: 0
--------------------------------------------------------------------
Decision Tree
A melhor nota:  54.644808743169406 %
melhor conjunto KFold: 0
--------------------------------------------------------------------
Naive Bayes
A melhor nota:  57.377049180327866 %
melhor conjunto KFold: 5
--------------------------------------------------------------------
Logistic Regression
A melhor nota:  57.14285714285714 %
A melhor fit_interception:  True
A melhor penalty:  0.3
melhor conjunto KFold: 7
--------------------------------------------------------------------
Random Forest
A melhor nota:  62.841530054644814 %
A melhor random_state:  64
A melhor max_depth:  8
A melhor n_estimators:  100
melhor conjunto KFold: 3
--------------------------------------------------------------------
o melhor modelo foi: Randon Forest
```

Fonte: Própria (2022)

Figura 12 – Histograma dos resultados dos treinamentos das IAs com os tweets de pessoas aleatórias



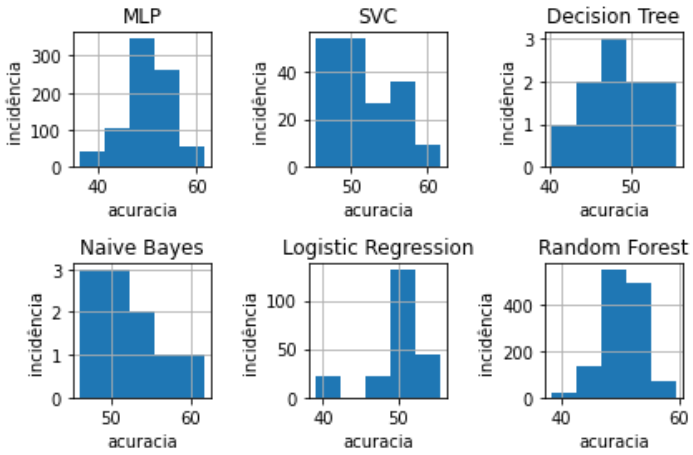
Fonte: Própria (2022)

Figura 13 – Resultados dos treinamentos das IAs com os tweets do Elon Musk

```
--------------------------------------------------------------------
MLP
A melhor nota:  61.702127659574465 %
melhor randon_state:  0
melhor hidden_layer_sizes:  1
melhor conjunto KFold: 5
--------------------------------------------------------------------
SVC
A melhor nota:  61.702127659574465 %
melhor randon_state:  0
melhor kernel:  rbf
melhor conjunto KFold: 5
--------------------------------------------------------------------
Decision Tree
A melhor nota:  57.04225352112676 %
melhor conjunto KFold: 0
--------------------------------------------------------------------
Naive Bayes
A melhor nota:  61.702127659574465 %
melhor conjunto KFold: 5
--------------------------------------------------------------------
Logistic Regression
A melhor nota:  55.633802816901415 %
A melhor fit_interception:  True
A melhor penalty:  0
melhor conjunto KFold: 0
--------------------------------------------------------------------
Random Forest
A melhor nota:  59.57446808510638 %
A melhor randon_state:  16
A melhor max_depth:  64
A melhor n_estimators:  100
melhor conjunto KFold: 8
--------------------------------------------------------------------
o melhor modelo foi: MLP
```

Fonte: Própria (2022)

Figura 14 – Histograma dos resultados dos treinamentos das IAs com os tweets do Elon Musk



Fonte: Própria (2022)

Apesar da baixa correlação no dataset do Elon Musk surpreendentemente os modelos conseguiram ter mais acurácia na predição do rumo das ações do que nos tweets aleatórios.

No tweet de anônimos o modelo que se sobressai é o de Random Forest, que obteve uma acurácia de 62,84%.

Já nos tweets do Elon Musk temos os modelos MLP, SVC e Naive Bayes com 61,70 % de acurácia porém temos uma incidência crescente no modelo MLP, então podemos dizer que ele trouxe o melhor resultado.

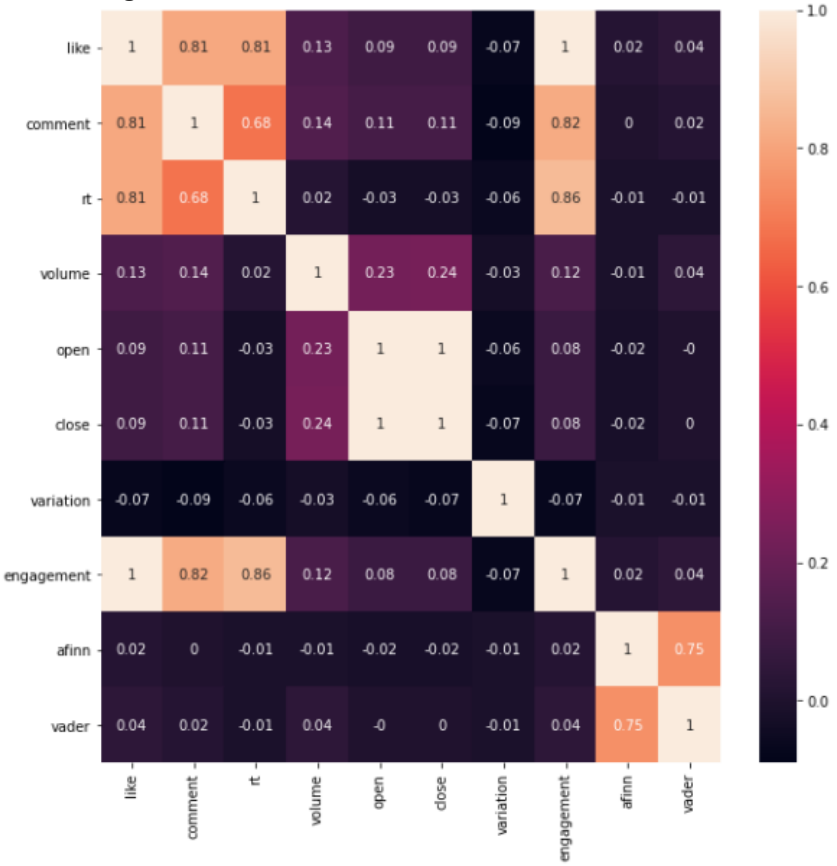
Após alguma discussão percebemos que poderíamos melhorar a performance do modelo filtrando os tweets que tinham alguma relação direta com a Tesla. Após a aplicação do filtro obtivemos os seguintes dados.

Figura 15 – Dados filtrados Elon Musk

	like	comment	rt	volume	open	close	variation	engagement	afinn	vader
count	918.0	918.0	918.0	918.0	918.0	918.0	918.0	918.0	918.0	918.0
mean	14198.0	480.0	1544.0	50436660.0	116.0	117.0	0.0	16221.0	0.0	1.0
std	28386.0	875.0	4342.0	33169451.0	129.0	130.0	2.0	32727.0	0.0	2.0
min	58.0	3.0	5.0	9362280.0	32.0	31.0	-8.0	70.0	-0.0	-4.0
25%	1828.0	92.0	109.0	27810466.0	50.0	50.0	-1.0	2277.0	0.0	0.0
50%	4853.0	198.0	369.0	40434785.0	63.0	63.0	0.0	5650.0	0.0	1.0
75%	14264.0	471.0	1327.0	65056312.0	98.0	96.0	1.0	15859.0	0.0	2.0
max	346163.0	12505.0	82934.0	304693800.0	675.0	666.0	9.0	384783.0	1.0	5.0

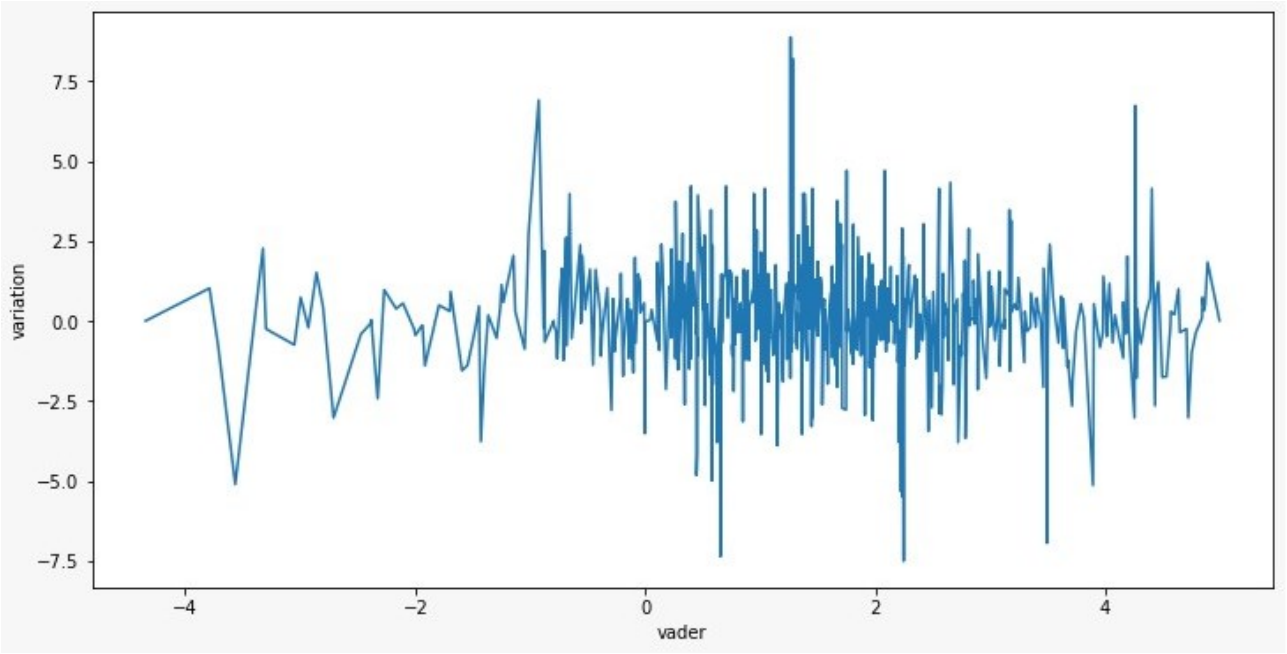
Fonte: Própria (2022)

Figura 16 – Correlação dataset Elon Musk filtrado



Fonte: Própria (2022)

Figura 17 – Variação das ações x Tweets do Elon Musk



Fonte: Própria (2022)

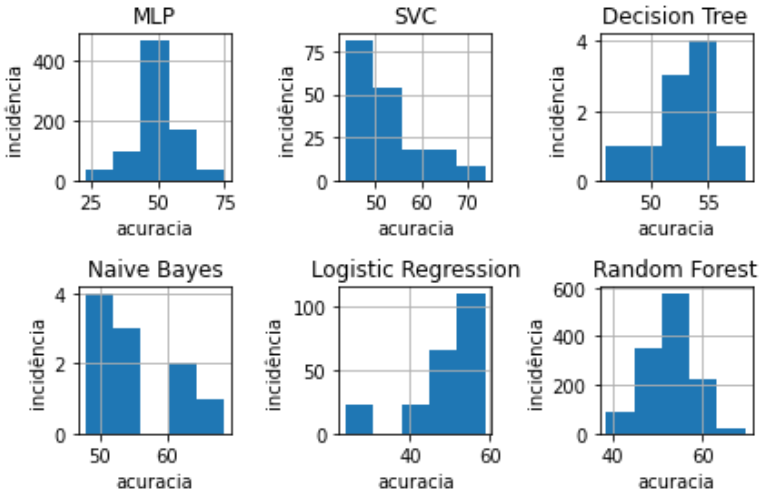
Apesar de termos diminuído a quantidade de dados, obtivemos resultados promissores incrementando a taxa de acurácia, como podemos observar no novo teste.

Figura 18 – Resultados dos treinamentos das IAs com os tweets do Elon Musk filtrados

```
----------------------------------------------------------------------
MLP
A melhor nota:  75.0 %
melhor randon_state:  0
melhor hidden_layer_sizes:  1
melhor conjunto KFold: 4
----------------------------------------------------------------------
SVC
A melhor nota:  73.91304347826086 %
melhor randon_state:  0
melhor kernel:  rbf
melhor conjunto KFold: 4
----------------------------------------------------------------------
Decision Tree
A melhor nota:  60.86956521739131 %
melhor conjunto KFold: 3
----------------------------------------------------------------------
Naive Bayes
A melhor nota:  68.47826086956522 %
melhor conjunto KFold: 4
----------------------------------------------------------------------
Logistic Regression
A melhor nota:  58.69565217391305 %
A melhor fit_interception:  True
A melhor penalty:  0
melhor conjunto KFold: 6
----------------------------------------------------------------------
Random Forest
A melhor nota:  69.56521739130434 %
A melhor randon_state:  2
A melhor max_depth:  2
A melhor n_estimators:  100
melhor conjunto KFold: 4
----------------------------------------------------------------------
o melhor modelo foi: MLP
```

Fonte: Própria (2022)

Figura 19 – Histograma dos resultados dos treinamentos das IAs com os tweets do Elon Musk filtrados



Fonte: Própria (2022)

Agora com os tweets filtrados por comentários relacionados a Tesla obtivemos uma taxa de 75% de acurácia no modelo MLP.

Esse resultado corrobora com nossa hipótese de que os tweets de Elon Musk podem e afetam os valores da ação de sua própria empresa,pela porcentagem de acurácia podemos também dizer que não somente suas declarações no Twitter impactam nos valores da ação mas também fatores externos.

5.Considerações Finais

Neste trabalho procuramos entender qual a influência e impacto das redes sociais no mercado financeiro, mais especificamente na relação dos tweets do Elon Musk com a variação das ações da sua empresa, Tesla. O objetivo é averiguar se podemos predizer quando uma ação vai valorizar ou desvalorizar de acordo com um tweet, e para fazê-lo optamos pela utilização de inteligência artificial para interpretar os dados.

Para alcançar o resultado esperado nós definimos dois objetivos. Primeiro analisamos o comportamento das ações nos baseando em tweets de pessoas anônimas, em sua maioria pessoas com baixo engajamento nas redes sociais. Nesse cenário obtivemos uma acurácia de 62,84% em predizer se ação iria subir ou descer o que não foi uma mal resultado considerando o cenário de pessoas anônimas. Já no nosso segundo objetivo que era analisar o mesmo comportamento mas com tweets do Elon Musk obtivemos uma acurácia de 61,70% o que é justificado pela baixa correlação das variáveis do dataset do Elon Musk.

Ao finalizarmos os dois objetivos percebemos que era possível melhorar a performance filtrando os tweets do Elon Musk por assuntos relacionados a Tesla, então definimos esse como um novo objetivo. Após a aplicação do filtro e os devidos ajustes conseguimos então nossa maior taxa de acurácia com 75%.

Com isso, a hipótese do trabalho de que o Elon Musk pode de alguma forma manipular o mercado financeiro ao seu bel prazer se confirmou, sendo corroborado pelos 75% de acurácia atingidos.

Sendo assim conseguimos atingir nosso principal objetivo que era relacionar a variação das ações da Tesla com os Tweets do seu dono Elon Musk, entretanto percebemos também

que as variações sofrem impactos externos a rede social e que não é o único fator que devemos levar em conta para tal variação.

Os resultados obtidos podem não refletir fidedignamente a realidade pois após inúmeros ajustes na massa de dados a quantidade de registros foi diminuída significativamente o que influencia diretamente no resultado mas não desvalida os resultados obtidos em nossa pesquisa.

Como melhoria do projeto pretendemos fazer uma integração com a API do Twitter para fazermos essa avaliação de forma automática e expandi-la para não só uma pessoa influente como o Elon Musk, mas qualquer veículo midiático e qualquer empresa presente na bolsa. Outra melhoria interessante proposta é implementar novos modelos de análise de sentimentos para sermos mais assertivos na classificação de um tweet.

6. Referências Bibliográficas

Internet
OPPENHEIMER, A. **How Does Social Media Influence Financial Markets?.** Disponível em: <https://www.nasdaq.com/articles/how-does-social-media-influence-financial-markets-2019-10-14> Acesso em: 09 fev. 2022

Internet
DUBOIS D. **The Most Influential CEOs on Twitter.** Disponível em: <https://knowledge.insead.edu/leadership-organisations/the-most-influential-ceos-on-twitter-4705> Acesso em: 9 fev. 2022

Internet
BIGGS, J. **Trump2Cash lets you invest automatically whenever the president mentions a publicly-traded company.** Disponível em: <https://techcrunch.com/2017/02/10/trump2cash-lets-you-invest-automatically-whenever-the-president-mentions-a-publicly-traded-company/> Acesso em: 09 fev. 2022

Internet
CAREY, N.; VENGATTIL, M. **U.S. judge approves SEC settlement with Tesla, Musk; shares jump.** Disponível em: <https://www.reuters.com/article/uk-tesla-musk-sec/u-s-judge-approves-sec-settlement-with-tesla-musk-shares-jump-idUSKCN1MQ20R> Acesso em: 9 fev. 2022

Internet
BRAUN, M. **This Machine Turns Trump Tweets into Planned Parenthood Donations.** Disponível em: <https://onezero.medium.com/this-machine-turns-trump-tweets-into-planned-parenthood-donations-4ece8301e722> Acesso em: 10 fev. 2022

Internet
ALI, F. **How Twitter affects the Stock Market.** Disponível em: <https://www.kaggle.com/code/faiqali1/how-twitter-affects-the-stock-market> Acesso em: 15 mar. 2022

Internet
HARJANTO, J. **Elon Musk & Twitter.** Disponível em: <https://towardsdatascience.com/elon-musk-twitter-adf324120b3f> Acesso em: 10 fev. 2022

Internet

YAMAMOTO, Y. **K417 - How Twitter affects the Stock Market.** Disponível em: <https://www.kaggle.com/code/hxtruong6/k417-how-twitter-affects-the-stock-market/notebook#Preprocessing> Acesso em: 15 mar. 2022

Artigo de Jornal
BLANKESPOOR, E.; MILLER, G.; WHITE, H. The Role of Dissemination in Market Liquidity: Evidence from Firms' Use of Twitter™. **The Accounting Review**, vol 89, no. 1, 2014, pp. 79–112. JSTOR, http://www.jstor.org/stable/24468513. Acesso em: 9 fev. 2022

Internet
CRUZ, D. V.; CORTEZ, V. F.; CHAU, A. L.; ALMAZAN, R. S.; **Does Twitter Affect Stock Market Decisions? Financial Sentiment Analysis During Pandemics: A Comparative Study of the H1N1 and the COVID-19 Periods.** Disponível em: <https://link.springer.com/article/10.1007/s12559-021-09819-8> Acesso em: 10 fev. 2022

Internet
STUMP, J. **Putting stock into Twitter: Social media can influence returns, WVU finance professor says.** Disponível em: <https://wvutoday.wvu.edu/stories/2020/11/09/putting-stock-into-twitter-social-media-can-influence-returns-wvu-finance-professor-says> Acesso em: 15 fev. 2022

Internet
Anônimo **How Seriously should investors consider the impact of Elon Musk's tweets on cryptos?** Disponível em: <https://www.livemint.com/market/cryptocurrency/how-seriously-should-investors-consider-the-impact-of-elon-musk-s-tweets-on-cryptos-11633589943118.html> Acesso em: 15 fev. 2022

Internet
VERCOE, P. **Seven Elon Musk Tweets That Sent Tesla Shares on a Wild Ride.** Disponível em: <https://www.bloomberg.com/news/articles/2021-11-09/seven-elon-musk-tweets-that-sent-tesla-shares-on-a-wild-ride> Acesso em: 15 fev. 2022

Internet
FISCHER, S. **How one Elon Musk tweet showed Twitter's impact on the stock market.** Disponível em: <https://www.axios.com/2022/05/14/elon-musk-tweet-twitter-tesla-stock-prices> Acesso em: 16 fev. 2022

Internet
McKinney, W.; PANDAS TEAM. **Pandas: powerful Python data analysis toolkit.** Disponível em: <https://pandas.pydata.org/docs/pandas.pdf> Acesso em: 10 mar. 2022

Internet
Hunter, J.; Dale, D.; Firing, E.; Droettboom M.; Matplotlib Development Team. **Matplotlib API reference** Disponível em: <https://matplotlib.org/stable/api/index.html> Acesso em: 5 mar. 2022

Internet
NumPy Community. **NumPy User Guide.** Disponível em: <https://numpy.org/doc/stable/numpy-user.pdf> Acesso em: 10 mar. 2022

Internet
Santiago, L. **Entendendo a biblioteca NumPy.** Disponível em: <https://medium.com/ensina-ai/entendendo-a-biblioteca-numpy-4858fde63355> Acesso em: 10 mar. 2022

Internet
WASKOM, M. **Seaborn API reference.** Disponível em: <https://seaborn.pydata.org/api.html> Acesso em: 15 mar. 2022

Internet
Nielsen, F. Å. **Afinn.** Disponível em: <https://github.com/fnielsen/afinn> Acesso em: 5 abr. 2022

Internet
Lohiya, H. **Sentiment Analysis with AFINN Lexicon.** Disponível em: <https://himanshulohiya.medium.com/sentiment-analysis-with-afinn-lexicon-930533dfe75b> Acesso em: 5 abr. 2022

Internet
Hutto, C.J. **VADER Sentiment Analysis.** Disponível em: <https://github.com/cjhutto/vaderSentiment> Acesso em: 5 abr. 2022

Internet
Thon, A. **Python Sentiment Analysis using VADER.** Disponível em: <https://www.geeksforgeeks.org/python-sentiment-analysis-using-vader/> Acesso em: 5 abr. 2022

Internet
Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; Vanderplas, J.; Passos, A.; Cournapeau, D.; Brucher, M.; Perrot, M.; Duchesnay, E. **scikit-learn Machine Learning in Python.** Disponível em: <https://scikit-learn.org/stable/> Acesso em: 15 abr. 2022

Internet
Lauro, B. **Algoritmo de Classificação Naive Bayes.** Disponível em: <https://www.organicadigital.com/blog/algoritmo-de-classificacao-naive-bayes/#:~:text=O%20algoritmo%20de%20Naive%20Bayes,dado%20que%20tem%20a%20doen%C3%A7a%E2%80%9D> Acesso em: 16 abr. 2022

Internet
Casali, R. **Como funciona o algoritmo Árvore de Decisão.** Disponível em: <https://www.digitalhouse.com/br/blog/arvore-de-decisao/> Acesso em: 16 abr. 2022

Internet
Navlani, A. **Understanding Random Forests Classifiers in Python Tutorial.** Disponível em: <https://www.datacamp.com/tutorial/random-forests-classifier-python> Acesso em: 16 abr. 2022

Internet

Lawton, G. **Logistic Regression** Disponível em: <https://www.techtarget.com/searchbusinessanalytics/definition/logistic-regression> Acesso em: 17 abr. 2022

Internet
Bento, C. **Multilayer Perceptron Explained with a Real-Life Example and Python Code: Sentiment Analysis.** Disponível em: <https://towardsdatascience.com/multilayer-perceptron-explained-with-a-real-life-example-and-python-code-sentiment-analysis-cb408ee93141> Acesso em: 17 abr. 2022

Internet
Moreira, S. **Rede Neural Perceptron Multicamadas.** Disponível em: <https://medium.com/ensina-ai/rede-neural-perceptron-multicamadas-f9de8471f1a9> Acesso em: 17 abr. 2022

Internet
Gandhi, G. **Support Vector Machine — Introduction to Machine Learning Algorithms.** Disponível em: <https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47> Acesso em: 17 abr. 2022

Artigo
Dhaoui, C.; Webster, C.M.; Tan, L.P. (2017), **Social media sentiment analysis: lexicon versus machine learning**, Journal of Consumer Marketing, Vol. 34 No. 6, p. 480-488.

Dissertação
FOTIOS, T. **Predicting Stocks Movement using Social Media Analytics.**Grecia, 47 p.,2018. tese (mestrado) - International Hellenic University

Dissertação
FILHO, P.L.M. **O Impacto nos Mercados Financeiros dos Tweets do Presidente Norte Americano Donald Trump.**Portugal, 42 p.,2020. tese (mestrado) - Escola Superior de Tecnologia e Gestão

Dissertação
Wang, F.; Hariandja, E.S. **The Influence of Brand Ambassador on Brand Image and Consumer Purchasing Decision: a Case Of Tous Les Joursin Indonesia**. Indonésia, 15 p., 2016. Dissertação – Universitas Pelita Harapan.

Artigo
Dhaoui, C.; Webster, C.M.; Tan, L.P. (2017), **Social media sentiment analysis: lexicon versus machine learning**, Journal of Consumer Marketing, Vol. 34 No. 6, p. 480-488.

ANEXOS

Segue em anexo os códigos referentes às partes principais do projeto após a organização dos dados que seriam respectivamente a análise de sentimentos, o filtro feito nos tweets do Elon Musk por palavras chave e o treinamento das inteligências artificiais para a tentativa de predizer a variação da bolsa. Para visualização completa do código de todas as etapas acesse o repositorio: https://github.com/VulcanoCorp/influence-stats

Anexo 1 - Análise de sentimentos

```
# instalação do afinn e o vader
!pip install afinn
!pip install vaderSentiment
# salvando os dataframe em variáveis
import pandas as pd
elon_data = pd.read_csv('data.csv')
data= pd.read_csv('tweets.csv')
# gerando a coluna engajamento
elon_data['engagement'] = elon_data['like'] + elon_data['comment'] +
elon_data['rt']
data['engagement'] = data['like'] + data['comment'] + data['rt']
# gerando a análise do afinn
from afinn import Afinn
afinn = Afinn()
data['afinn'] = data['tweet'].apply(lambda tweet: afinn.score(tweet))/20;
elon_data['afinn'] = elon_data['tweet'].apply(lambda tweet:
afafinn.score(tweet))/20;
# algoritmo para a nota do vader
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer
vader = SentimentIntensityAnalyzer()
def getVaderScore(tweet):
    vader_score = vader.polarity_scores(tweet)
    score = vader_score['compound']
    return score*5inn.score(tweet))/20
```

```
# dando a nota do vader para os textos
data['vader'] = data['tweet'].apply(lambda tweet: getVaderScore(tweet))
elon_data['vader'] = elon_data['tweet'].apply(lambda tweet:
getVaderScore(tweet))
```

Anexo 2 - Filtrando os dados do Elon Musk por texto

```
#gerando novo data frame com os tweets do elon musk filtrados por
palavras chave

from re import X
from datetime import datetime, timedelta
import numpy as np
elon_tweets= elon_data['tweet'].tolist()
disorder = True
cicles = 1
output = []
for x, item in enumerate(elon_tweets):
  if 'tesla' in item:
    output.append(x)
  elif 'car' in item:
    output.append(x)
  elif 'model 3' in item:
    output.append(x)
  elif 'model s' in item:
    output.append(x)
  elif 'model x' in item:
    output.append(x)
  elif 'model y' in item:
    output.append(x)
  elif 'tsla' in item:
    output.append(x)
  elif 'eletric' in item:
    output.append(x)
  elif 'solar' in item:
    output.append(x)
elon_filtered_data = elon_data.loc[output]
```

Anexo 3 - Treinamento e testes das Inteligências Artificiais

```
# algoritmo de grid search e cross validation
from sklearn.model_selection import cross_val_score
from sklearn.neural_network import MLPClassifier
from sklearn.model_selection import KFold
from sklearn.tree import DecisionTreeClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn import tree
from sklearn import svm
import sys

def best_model(data,result,Models_tested = 'A'):
  output_results = [];  output_groups = [];
  randon_range = [0,1,2,4,8,16,32,64,128];
  depth_range = [2,4,8,16,32,64,128]; tree_range = [100, 500];
  hidden_range = [100,1,2,4,8,16,32,64,128]
  penalty_options = [ 0,0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1];
  kernel_range = ['rbf', 'sigmoid']; intercept_options = [True, False];
  kf = KFold(n_splits=10); best_kernel = ''; best_intercept = True;
  best_randon = 0; best_hidden = 0; best_train = []; best_1=0; best_2=0;
  best_3=0; best_4=0; best_5=0; best_6=0; best_score = 0; best_depth = 0;
  best_tree = 0; best_test = []; best_model = '';
  best_penalty= ''; kfsplit = 0; best_kf = 0

  if 'A' in Models_tested:
    for randon in randon_range:
      for hidden in hidden_range:
        kfsplit = 0
        for train_index, test_index in kf.split(data):
            X_train, X_test = data.loc[train_index], data.loc[test_index]
            Y_train, Y_test = result[train_index], result[test_index]
            clf = MLPClassifier(random_state=randon,
hidden_layer_sizes=hidden, max_iter=1000)
```

```
            clf.fit(X_train,Y_train)
            score = clf.score(X_test, Y_test)*100
            output_groups.append(score)
            if(score > best_1):
              best_train = train_index
              best_test = test_index
              best_1 = score
              best_randon = randon
              best_hidden = hidden
              best_kf = kfsplit
            sys.stdout.write('\r'+ f"atual-> {score}% kfold->{kfsplit}
melhor-> {best_1}%");
            kfsplit = kfsplit+1

    sys.stdout.write('\r'+ '----------------------------------------')

    print('')
    print('MLP')
    print('A melhor nota: ', best_1,'%')
    print('melhor randon_state: ',best_randon)
    print('melhor hidden_layer_sizes: ',best_hidden)
    print('melhor conjunto KFold:', best_kf)
    output_results.append(output_groups)
    output_groups = []
    best_score = best_1
    best_randon = 0
    best_model = 'MLP'
  if 'B' in Models_tested :
    print('0%', end="")
    for randon in randon_range:
      for kernel in kernel_range:
        kfsplit = 0
        for train_index, test_index in kf.split(data):
            X_train, X_test = data.loc[train_index], data.loc[test_index]
            Y_train, Y_test = result[train_index], result[test_index]
            clf = svm.SVC(kernel=kernel, C=1, random_state=randon)
            clf.fit(X_train,Y_train)
            score = clf.score(X_test, Y_test)*100
            output_groups.append(score)
            if(score > best_2):
              best_2 = score
              best_randon = randon
              best_kernel = kernel
              best_kf = kfsplit
              if best_2 > best_score:
                best_score = best_2
                best_train = train_index
                best_test = test_index
                best_model = 'SVC'
            sys.stdout.write('\r'+ f"atual-> {score}% kfold->{kfsplit}
melhor-> {best_2}%");
            kfsplit = kfsplit+1

    sys.stdout.write('\r'+ '----------------------------------------')
    print('SVC')
    print('A melhor nota: ', best_2,'%')
    print('melhor randon_state: ',best_randon)
    print('melhor kernel: ',best_kernel)
    print('melhor conjunto KFold:', best_kf)
```

```
  output_results.append(output_groups)
  output_groups = []
if 'C' in Models_tested :
  print('0%', end="")
  kfsplit = 0
  for train_index, test_index in kf.split(data):
    X_train, X_test = data.loc[train_index], data.loc[test_index]
    Y_train, Y_test = result[train_index], result[test_index]
    clf = DecisionTreeClassifier()
    clf.fit(X_train,Y_train)
    score = clf.score(X_test, Y_test)*100
    output_groups.append(score)
    if(score > best_3):
      best_3 = score
      best_kf = kfsplit
      if best_3 > best_score:
        best_score = best_3
        best_train = train_index
        best_test = test_index
        best_model = 'Decision Tree'
    sys.stdout.write('\r'+ f"atual-> {score}% melhor-> {best_3}%")
    kfsplit = kfsplit+1

  sys.stdout.write('\r'+ '----------------------------------------')
  print('Decision Tree')
  print('A melhor nota: ', best_3,'%')
  print('melhor conjunto KFold:', best_kf)
  output_results.append(output_groups)
  output_groups = []
if 'D' in Models_tested :
  print('0%', end="")
  kfsplit = 0
  for train_index, test_index in kf.split(data):
    X_train, X_test = data.loc[train_index], data.loc[test_index]
    Y_train, Y_test = result[train_index], result[test_index]
    clf = GaussianNB()
    clf.fit(X_train,Y_train)
    score = clf.score(X_test, Y_test)*100
    output_groups.append(score)
    if(score > best_4):
      best_4 = score
      best_kf = kfsplit
      if best_4 > best_score:
        best_score = best_4
        best_train = train_index
        best_test = test_index
        best_model = 'Naive Bayes'
    sys.stdout.write('\r'+ f"atual-> {score}% melhor-> {best_4}%")
    kfsplit = kfsplit+1
  sys.stdout.write('\r'+ '----------------------------------------')
  print('Naive Bayes')
  print('A melhor nota: ', best_4,'%')
  print('melhor conjunto KFold:', best_kf)
  output_results.append(output_groups)
  output_groups = []
if 'E' in Models_tested :
```

```
    print('0%', end="")
    for option in intercept_options:
      for penalty in penalty_options:
        kfsplit = 0
        for train_index, test_index in kf.split(data):
          X_train, X_test = data.loc[train_index], data.loc[test_index]
          Y_train, Y_test = result[train_index], result[test_index]
      clf = LogisticRegression(fit_intercept=option,
penalty='elasticnet', solver='saga', l1_ratio=penalty, max_iter=10000);
          clf.fit(X_train,Y_train)
          score = clf.score(X_test, Y_test)*100
          output_groups.append(score)
          if(score > best_5):
            best_5 = score
            best_intercept = option
            best_penalty = penalty
            best_kf = kfsplit
            if best_5 > best_score:
              best_score = best_5
              best_train = train_index
              best_test = test_index
              best_model = 'Logistic Regression'
            sys.stdout.write('\r'+ f"atual-> {score}% penalty->{penalty}
melhor-> {best_5}%");
          kfsplit = kfsplit+1
    sys.stdout.write('\r'+ '-----------------------------------------')
    print('Logistic Regression')
    print('A melhor nota: ', best_5,'%')
    print('A melhor fit_interception: ', best_intercept)
    print('A melhor penalty: ', best_penalty)
    print('melhor conjunto KFold:', best_kf)
    output_results.append(output_groups)
    output_groups = []
  if 'F' in Models_tested :
    print('0%', end="")
    for tree in tree_range:
      for randon in randon_range:
        for depth in depth_range:
          kfsplit = 0
          for train_index, test_index in kf.split(data):
            X_train, X_test = data.loc[train_index], data.loc[test_index]
            Y_train, Y_test = result[train_index], result[test_index]
            clf = RandomForestClassifier(n_estimators= tree,
max_depth=depth, random_state=randon,);
            clf.fit(X_train,Y_train)
            score = clf.score(X_test, Y_test)*100
            output_groups.append(score)
            if(score > best_6):
              best_6 = score
              best_randon = randon
              best_tree = tree
              best_depth = depth
              best_kf = kfsplit
              if best_6 > best_score:
                best_score = best_6
                best_train = train_index
                best_test = test_index
```

```
                best_model = 'Randon Forest'
              sys.stdout.write('\r'+ f"atual-> {score}% depth->{depth}
melhor-> {best_6}%");
            kfsplit = kfsplit+1

    sys.stdout.write('\r'+ '----------------------------------------')
    print('Random Forest')
    print('A melhor nota: ', best_6,'%')
    print('A melhor randon_state: ', best_randon)
    print('A melhor max_depth: ', best_depth)
    print('A melhor n_estimators: ', best_tree)
    print('melhor conjunto KFold:', best_kf)
    output_results.append(output_groups)
    output_groups = []
print('------------------------------------------------------------')
  print(f'o melhor modelo foi: {best_model}')
  print("\n\n")
  return output_results



# algoritmo para plotagem dos histogramas dos resultados da IA
import matplotlib.pyplot as plt
import pandas as pd
def grid_hist(hist_data):
  models = ["MLP","SVC","Decision Tree","Naive Bayes", "Logistic
Regression", "Random Forest"]
  fig=plt.figure()
  for idx,item in enumerate(hist_data):
    ax=fig.add_subplot(2,3,idx+1)
    data={'values': item}
    df = pd.DataFrame(data)
    df['values'].hist(bins=5,ax=ax)
    ax.set_title(models[idx])
  fig.tight_layout()
  plt.show()
# treinamento das IAs com os dados dos tweets de pessoas genéricas
geral_results = best_model(X,Y,Models_tested = 'A B C D E F')
grid_hist(geral_results)

# treinamento das IAs com os dados dos tweets do elon musk
elon_results = best_model(elon_X,elon_Y,Models_tested = 'A B C D E F')
grid_hist(elon_results)

# treinamento das IAs com os dados dos tweets do elon musk filtrados
filtered_results =  best_model(filtered_X,filtered_Y,Models_tested = 'A B
C D E F');
grid_hist(filtered_results)
```