## Desenvolvimento

Neste documento estaremos explicando sobre os detalhes do projeto na área de IA da Sprint 2, porém, pra isso é necessário entender a primeira sprint.

Na primeira sprint, definimos o macro de todas as etapas, incluindo a parte de Inteligência Artificial. Decidimos que será necessário o tratamento de dados históricos provenientes de redes sociais, softwares de tráfego pago e outras fontes de informações na internet. No entanto, surgiram dúvidas sobre as tecnologias necessárias para a criação do modelo de aprendizado com análise profunda (IA & Deep Analytics), os tipos de dados de entrada e como deveríamos tratá-los.

## IA & Deep Analytics

Nossa solução requer uma análise profunda e abrangente de uma grande base de dados históricos para entender os padrões e tendências de comportamento dos consumidores ao longo do tempo. Em vez de uma equipe de pessoas, utilizaremos um modelo de aprendizado de máquina, uma IA. Isso reduzirá exponencialmente o tempo necessário e diminuirá a taxa de erro humano, pois a IA será capaz de lidar com grandes volumes de dados de maneira eficiente. Dessa forma, asseguraremos que os resultados obtidos sejam precisos e confiáveis, permitindo que as decisões de campanha sejam baseadas em informações sólidas.

## Tipo e Tratamento de Dados

Dados históricos serão extraídos por meio de web scraping de sites, redes sociais e softwares de tráfego pago. Esses dados incluirão campos como data de nascimento, para determinar a idade dos usuários; comentários e descrições, para identificar hashtags, tópicos e discussões relevantes; e a fonte, que indica a origem dos dados, como a rede social de onde o comentário foi extraído, entre outros metadados.

Os dados serão armazenados em um banco de dados e utilizados na análise do modelo de inteligência artificial. Isso gerará valores numéricos compreensíveis para o analista de marketing digital, permitindo a criação de campanhas publicitárias de maneira mais eficiente, economizando tempo, recursos e trabalho.

## Bibliotecas

TensorFlow surgiu a partir do DisBelief, um framework de código fechado do Google, e foi lançado pela primeira vez em 2015 sob a licença Apache 2.0, tornando-se um framework de código aberto. Essa abertura impulsionou seu desenvolvimento devido aos feedbacks dos usuários. Atualmente, é o framework de deep learning mais famoso e, por exemplo, usa-se a API Keras, uma biblioteca que simplifica o treinamento de novos modelos.

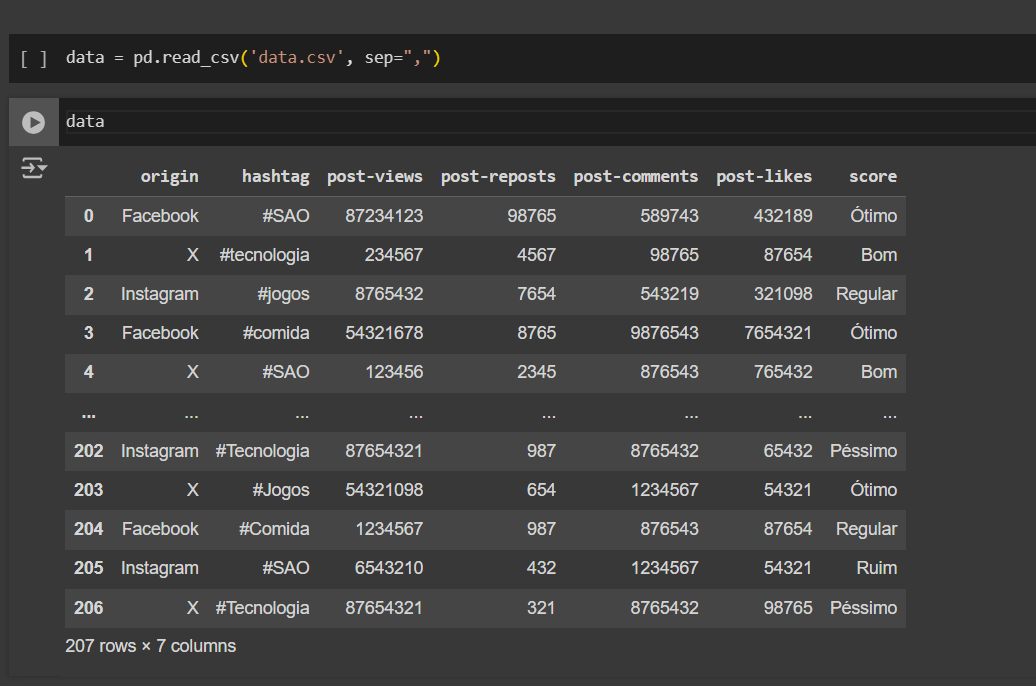
O TensorFlow possui todas as características necessárias para a VOM-Hive: confiança, por ser uma ferramenta estabelecida no mercado e desenvolvida pela Google; praticidade, por estar amplamente documentada e ser de código aberto, facilitando o aprendizado e a utilização; e, por fim, flexibilidade, pois pode ser utilizada em diversas linguagens como Python, Java e C++.

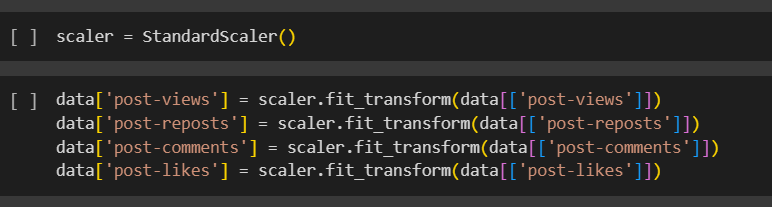
## 

## Beta

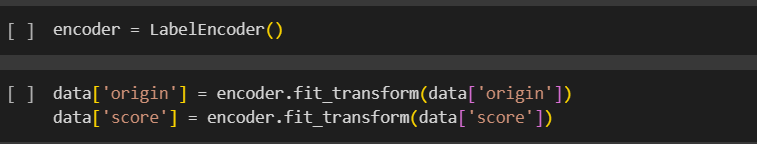
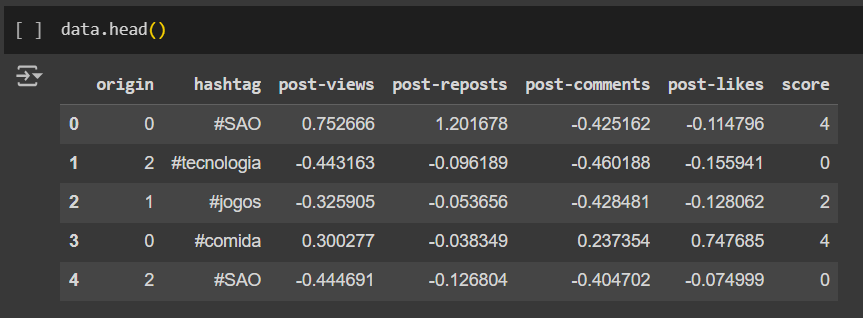
Antes de explicar o beta do projeto, é importante entender que ele serve apenas para demonstrar a ideia do resultado esperado da Inteligência Artificial. Os dados utilizados não são reais, mas sim exemplos gerados por IA generativa exclusivamente para fins de demonstração. Segundo o site Futurism, que cita um estudo de cientistas da Universidade Rice e da Universidade de Stanford, alimentar modelos de IA com conteúdo gerado por outras IAs resulta na deterioração da qualidade dos resultados desses modelos. Portanto, não se espera que os resultados do beta sejam precisos ou de alta qualidade.

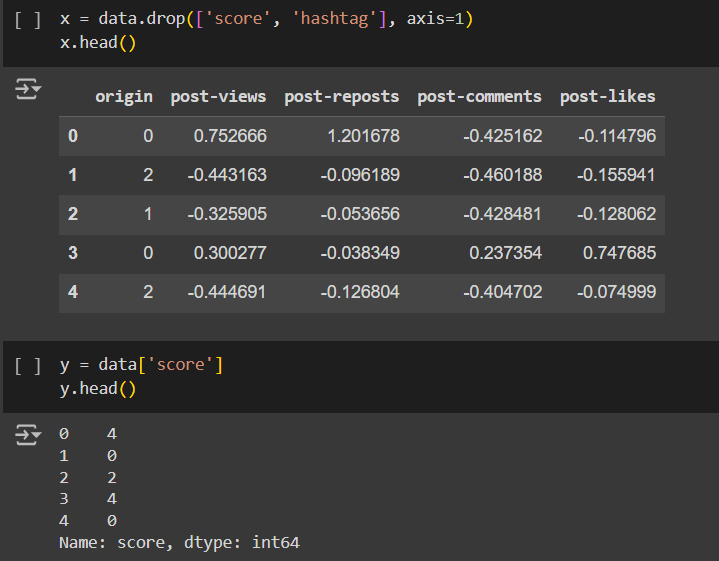
O modelo foi feito treinado com o dataset no seguinte formato:

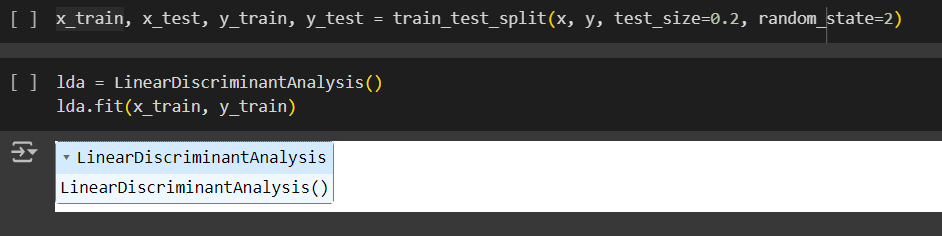


Para o treinamento usamos o [StandardScaler](https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html#sklearn.preprocessing.StandardScaler) do SKLearn para padronizar os dados numéricos.

Em seguida usamos o [LabelEncoder](https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.LabelEncoder.html) do SKLearn utilizando da técnica de OneHotEncoding.

Demonstração dos dados após a manipulação:

Na sequência fizemos a separação de entrada e da coluna target.

Com a separação dos dados de entrada e da coluna target, usamos o [TrainTestSplit](https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html#sklearn.model_selection.train_test_split) do SKLearn, de dados de treino e teste. Com os dados separados Criamos o modelo de aprendiziado de máquina de classificação com o [LinearDiscriminantAnalysis](https://scikit-learn.org/stable/modules/generated/sklearn.discriminant_analysis.LinearDiscriminantAnalysis.html#sklearn.discriminant_analysis.LinearDiscriminantAnalysis) do SKLearn.

Por fim, o testo do modelo e suas métricas:

