Relatório Projeto Spark
Megadados

Professor Fábio Ayres

Eiki Luis Yamashiro
João Guilherme Cintra de Freitas Almeida
William Augusto Reis da Silva

1. Introdução

O objetivo do projeto é analisar um dataset que contém reviews de produtos da Amazon [2], através do Spark [1]. As colunas que compõem o dataset são as seguintes:

marketplace: Código do País de origem do marketplace do review (2 letras)
customer_id: Identificador aleatório que é utilizado para identificar um único autor.
review_id: Identificador único do review.
product_id: O ID exclusivo do produto ao qual a revisão pertence. No conjunto de dados multilíngue, as análises para o mesmo produto em diferentes países podem ser agrupadas pelo mesmo product_id.
product_parente: Identificador aleatório que pode ser usado para agregar avaliações para o mesmo produto.
product_title: Título do produto.
product_category: Categoria de produto que pode ser usada para agrupar avaliações.
star_rating: A classificação de 1 a 5 estrelas da avaliação.
helpful_votes: Número de votos úteis.
total_votes: Número total de votos que a revisão recebeu.
vine: A revisão foi escrita como parte do programa Vine.
verified_purchase: A revisão é de uma compra verificada.
review_headline: O título da revisão.
review_body: O texto da revisão.
review_date: A data em que a revisão foi escrita.

Os dados estão formatados em tsv, ou seja, são delimitados por Tab (‘\t’).

2. Task 1

a. **Quantos reviews existem?**
   Utilizando a quantidade de linhas que possuía no dataset, obtemos o valor de 150.962.278 reviews.

b. **Quantos clientes existem?**

   Utilizando a quantidade de linhas que possuía na coluna “customer_id”, pegando somente os valores diferentes, obtemos o valor de 33.497.620 clientes.

c. **Quantos produtos existem?**

   Utilizando a quantidade de linhas que possuía na coluna “product_id”, pegando somente os valores diferentes, obtemos o valor de 21.390.118 produtos.

d. **Quantos reviews existem para cada “star_rating”?**
   Utilizando um groupBy(“star_rating”) e filtrando somente os que possuíam valores de 1 a 5, tendo em vista que parecia ter um bug com datas em alguns locais, obtivemos os seguintes resultados:

star_rating	*quantidade*
1	12.099.424
2	7.304.329
3	12.133.772
4	26.223.155
5	93.199.322

Tabela 1

3. Task 2

Para considerar um usuário como um bot, será feita uma categorização a qual todo o usuário que avaliar mais de duas vezes um produto será classificado como um bot. Isso pois bots são aqueles que são usados para interferir na avaliação de um produto, dessa forma, se um usuário avalia mais de 2 vezes um produto, o provável motivo disso é que este quer interferir na nota do produto, seja para piorá-lo dando sempre 1 ou melhorá-lo dando notas altas.

Com os resultados, foi possível descobrir que a fonte de dados tinha 35549 bots (0,1% dos clientes), os quais boa parte faziam reviews para as categorias Books, Music, Home e Video Games. Devido à forma como os bots foram classificados a categoria de produtos ”Grocery” acompanhou alguns bots. Ao tentar interpretar o resultado, concluímos que talvez não faz sentido considerar como bot, pois é comum que pessoas comprem em mercados (Grocery) mais de duas vezes e avalie.

Os outros, por exemplo, imaginamos que realmente sejam bots. Ainda mais no ramo dos livros, que inclui e-books, onde há muita competição e diversos artistas independentes que necessitam que os livros possuam uma relevância maior dentro da loja para atrair mais clientes. Tendo mais avaliações, ele acaba sendo visto como mais relevante.

Além de tudo isso, também observa-se que a maior parte das notas são positivas, o que demonstra que o principal objetivo é fazer com que o produto cresça de relevância na plataforma para angariar mais vendas com suas avaliações positivas.

4. Task 3

Na tarefa três é solicitado a construção de um classificador naive-Bayes para determinar se o review é positivo, neutro ou negativo. A definição da métrica é apresentada a seguir:

5 estrelas → positivo
4 estrelas → neutro
3 ou menos estrelas → negativo

O primeiro passo para a construção é o tratamento dos dados, o classificador não aceitará valores de star_rating nulos, portanto é necessário retirá-los do conjunto. Existem

também alguns dados em star_rating que não seguem a definição da coluna (há algumas datas no meio da coluna), assim, também será necessário excluir esses dados do dataset.

O segundo passo, é a definição da métrica, como apresentado acima. Dessa forma, seguindo as limitações impostas pela métrica, cria-se uma nova coluna de metric (string), que pode assumir os seguintes valores: “Positive”, “Neutro” ou “Negative”. Com a remoção ds dados da primeira etapa e a criação da coluna metrics, o conjunto de dados está pronto para o treinamento. A seguir é apresentado uma tabela que representa um recorte do conjunto:

review_body	star_rating	metric
Dyan Cannon, the ...	4	Neutro
The book was in e...	5	Positive
This book deals w...	3	Negativo

Tabela 2 - Estrutura do Conjunto Tratado para o Naive-Bayes

A próxima etapa consiste em normalizar as letras maiúsculas e minúsculas, dividir as frases em uma lista de palavras, através do RegexTokenizer.

Em seguida, o Count Vectorizer converte os dados de texto em um vetor de contagem de tokens (termos). A coluna de rótulo (metric) é convertida de strings para rótulos numéricos. Ao final todos os recursos são juntados em um único vetor através do VectorAssembler. Após a construção do vetor, o pipeline é feito seguido da divisão em conjunto de treino e teste, que no caso foi feito com uma proporção de 70% para treino e 30% para teste.

Assim, basta implementar o Naive-Bayes e treinar o modelo. Para a avaliação do modelo, utilizaremos um BinaryClassificationEvaluator do pacote de machine learning do pyspark. A acurácia obtida do modelo foi de 0.7417081742136926.

Referências

[1] Unified engine for large-scale data analytics, Apache Spark. Disponível em https://spark.apache.org/ . Acesso em 10/12/2021.

[2] Amazon Customer Reviews Dataset. Disponível em https://s3.amazonaws.com/amazon-reviews-pds/readme.html . Acesso em 10/12/2021.