Title: Exploratory Study of TF-IDF Text Feature Extraction in Apache Spark for Big Data Text Mining
Task Type: Literature Review with Code Analysis and EDA
Overview: As my task is targeted on reviewing huge-scale textual content mining in disbursed environments, I explored a realistic evidence-of-concept the use of Apache Spark's MLlib library to process and extract functions from textual content records the use of the TF-IDF (Term Frequency-Inverse Document Frequency) algorithm.
This was done the usage of pattern Spark code (linked underneath), wherein a listing of documents turned into converted into time period vectors and TF-IDF scores using distributed processing.
Steps Taken:

- Installed Apache Spark and PySpark
- Used a small dummy dataset simulating a large corpus (3–5 documents)
- Applied Tokenizer, HashingTF, and IDF from pyspark.Ml.Function
- Extracted top TF-IDF rankings according to report

Code Source:
https://spark.apache.org/docs/latest/ml-features.html#tf-idf-term-frequency--inverse-document-frequency
Modified Sample Code Source:
https://colab.research.google.com/drive/1owDTB4flnT3Ot0K3lflgO4sg8lq2uelw?usp=sharing
The usage of Spark ML Pipeline:

- Tokenization of words
- Stopwords removed
- Bigrams generated (2-word terms)
- TF-IDF capabilities computed

Visualizations Created:

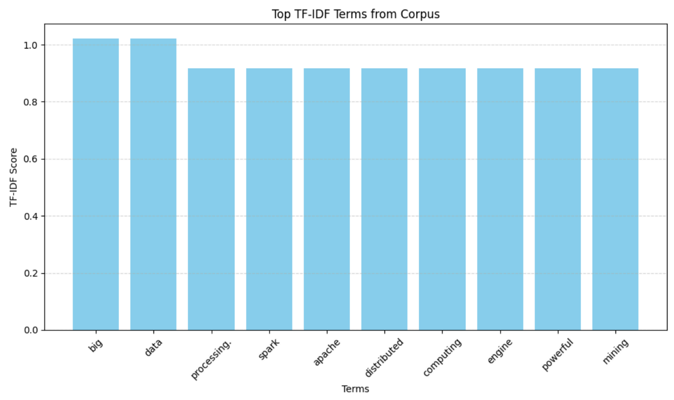

Fig: Bar Chart

Conclusion: This test demonstrates how distributed textual content mining pipelines can be applied and analyzed the use of Apache Spark. Even with a small dataset, I become capable of simulate actual-world preprocessing and advantage insights the usage of EDA. This supports my scientific literature assessment.