SPARK İLE COLAB ÜZERİNDEN sosyal ağ duygu analizi ile ilgili çalışan bir kod örneği

Denemsi

# Gerekli Kütüphaneleri Yükleme

!apt-get install openjdk-8-jdk-headless -qq > /dev/null

!wget -q https://archive.apache.org/dist/spark/spark-3.2.0/spark-3.2.0-bin-hadoop3.2.tgz

!tar xf spark-3.2.0-bin-hadoop3.2.tgz

!pip install -q findspark

import os

os.environ["JAVA\_HOME"] = "/usr/lib/jvm/java-8-openjdk-amd64"

os.environ["SPARK\_HOME"] = "/content/spark-3.2.0-bin-hadoop3.2"

import findspark

findspark.init()

# Spark'ı Başlatma

from pyspark.sql import SparkSession

try:

    spark = SparkSession.builder \

        .appName("Sentiment Analysis") \

        .getOrCreate()

except Exception as e:

    print(f"Spark başlatılırken hata oluştu: {e}")

# Veri Setini Yükleme

try:

    # Geçerli bir veri seti URL'si (alternatif olarak Kaggle veri seti yükleyebilirsiniz)

    !wget -O tweets.csv https://raw.githubusercontent.com/dD2405/Twitter\_Sentiment\_Analysis/master/train.csv

    df = spark.read.csv("tweets.csv", header=True, inferSchema=True)

    df.show(5)

except Exception as e:

    print(f"Veri seti yüklenirken hata oluştu: {e}")

# Veri Ön İşleme

from pyspark.sql.functions import col, lower, regexp\_replace

try:

    # Gereksiz sütunları kaldırma

    df = df.select("tweet", "label")  # "tweet" ve "label" sütunlarını kullanın (tweet metni ve duygu etiketi)

    # Metni küçük harfe çevirme ve özel karakterleri kaldırma

    df = df.withColumn("tweet", lower(col("tweet")))

    df = df.withColumn("tweet", regexp\_replace(col("tweet"), "[^a-zA-Z\\s]", ""))

    df.show(5)

except Exception as e:

    print(f"Veri ön işleme sırasında hata oluştu: {e}")

# Özellik Çıkarımı

from pyspark.ml.feature import Tokenizer, HashingTF, IDF

try:

    # Kelimeleri tokenize etme

    tokenizer = Tokenizer(inputCol="tweet", outputCol="words")

    words\_data = tokenizer.transform(df)

    # HashingTF kullanarak kelimeleri sayısallaştırma

    hashing\_tf = HashingTF(inputCol="words", outputCol="rawFeatures", numFeatures=100)

    featurized\_data = hashing\_tf.transform(words\_data)

    # IDF ile TF-IDF hesaplama

    idf = IDF(inputCol="rawFeatures", outputCol="features")

    idf\_model = idf.fit(featurized\_data)

    tfidf\_data = idf\_model.transform(featurized\_data)

except Exception as e:

    print(f"Özellik çıkarımı sırasında hata oluştu: {e}")

# Model Eğitimi

from pyspark.ml.classification import NaiveBayes

try:

    # Modeli eğitme

    nb = NaiveBayes(featuresCol='features', labelCol='label')

    model = nb.fit(tfidf\_data)

    # Modelin performansını değerlendirme

    predictions = model.transform(tfidf\_data)

    predictions.select("tweet", "label", "prediction").show(5)

except Exception as e:

    print(f"Model eğitimi sırasında hata oluştu: {e}")

# Sonuçları Değerlendirme

from pyspark.ml.evaluation import MulticlassClassificationEvaluator

try:

    evaluator = MulticlassClassificationEvaluator(labelCol="label", predictionCol="prediction", metricName="accuracy")

    accuracy = evaluator.evaluate(predictions)

    print(f"Model Doğruluğu: {accuracy:.2f}")

except Exception as e:

    print(f"Sonuç değerlendirme sırasında hata oluştu: {e}")