Netflix Content-Based Recommendation System using PySpark

This notebook demonstrates how to build a recommendation system that suggests similar Netflix shows or movies based solely on their content (title, genre, and description). It uses PySpark for large-scale data processing and applies tokenization, stopword removal, and a custom Jaccard similarity function to identify the most similar titles without relying on user ratings or collaborative filtering techniques.

```
# Step 1: Load the Netflix dataset
netflix_df = (
    spark.read.option("header", True)
               .option("inferSchema", True)
               .csv("/FileStore/netflix_titles.csv")
)

# Clean null values and combine 'title', 'listed_in', and 'description' into a single lowercase text column
from pyspark.sql.functions import col, concat_ws, lower

netflix_df = (
    netflix_df.na.fill("")
              .withColumn("text", concat_ws(" ",
                                            lower(col("title")),
                                            lower(col("listed_in")),
                                            lower(col("description"))))
)

# Show first 5 rows of cleaned data
netflix_df.select("title", "text").show(5, truncate=False)
```

```
+---------------------+----------------------------------------------------------------------------------------------------------------------------
|title                |text
+---------------------+----------------------------------------------------------------------------------------------------------------------------
|Dick Johnson Is Dead |dick johnson is dead documentaries as her father nears the end of his life, filmmaker kirsten johnson stages
|Blood & Water        |blood & water international tv shows, tv dramas, tv mysteries after crossing paths at a party, a cape town te
|Ganglands            |ganglands crime tv shows, international tv shows, tv action & adventure to protect his family from a powerful
|Jailbirds New Orleans|jailbirds new orleans docuseries, reality tv feuds, flirtations and toilet talk go down among the incarcerate
|Kota Factory         |kota factory international tv shows, romantic tv shows, tv comedies in a city of coaching centers known to tr
+---------------------+----------------------------------------------------------------------------------------------------------------------------
only showing top 5 rows
```

```
# Step 2: Tokenize the text and remove common stopwords
from pyspark.ml.feature import Tokenizer, StopWordsRemover

# Tokenizer splits text into words
tokenizer = Tokenizer(inputCol="text", outputCol="words")

# StopWordsRemover filters out common English words
stopper = StopWordsRemover(inputCol="words", outputCol="filtered_words")

# Apply tokenizer
tokenized = tokenizer.transform(netflix_df)

# Apply stopword remover
processed = stopper.transform(tokenized).select("title", "filtered_words").cache()

# Show sample of processed data
processed.show(5, truncate=False)
```

```
+---------------------+----------------------------------------------------------------------------------------------------------------------------
|title                |filtered_words
+---------------------+----------------------------------------------------------------------------------------------------------------------------
|Dick Johnson Is Dead |[dick, johnson, dead, documentaries, father, nears, end, life,, filmmaker, kirsten, johnson, stages, death, i
|Blood & Water        |[blood, &, water, international, tv, shows,, tv, dramas,, tv, mysteries, crossing, paths, party,, cape, town,
|Ganglands            |[ganglands, crime, tv, shows,, international, tv, shows,, tv, action, &, adventure, protect, family, powerful
|Jailbirds New Orleans|[jailbirds, new, orleans, docuseries,, reality, tv, feuds,, flirtations, toilet, talk, go, among, incarcerate
|Kota Factory         |[kota, factory, international, tv, shows,, romantic, tv, shows,, tv, comedies, city, coaching, centers, known
+---------------------+----------------------------------------------------------------------------------------------------------------------------
only showing top 5 rows
```

```
# Step 3: Define Jaccard similarity function using UDF
from pyspark.sql.functions import udf
from pyspark.sql.types import DoubleType

# Function to compute Jaccard similarity between two lists
```

```
def jaccard_similarity(list1, list2):
    set1, set2 = set(list1), set(list2)
    if not set1 and not set2:
        return 0.0
    return float(len(set1 & set2)) / len(set1 | set2)

# Register it as a Spark UDF
jaccard_udf = udf(jaccard_similarity, DoubleType())



# Step 4: Define the recommend function
def recommend(title: str, num_recs: int = 5):
    # Find the row with the input title (case-insensitive)
    title_row = processed.filter(lower(col("title")) == title.lower()).collect()

    # If title not found, print and return
    if not title_row:
        print(f"No such title found: '{title}'")
        return

    # Extract the filtered_words (token list) of the selected title
    tokens = title_row[0]['filtered_words']
    actual_title = title_row[0]['title']
    print(f"Target Title: {actual_title}")

    # Create a one-row DataFrame for the target token list
    input_df = spark.createDataFrame([(tokens,)], ["filtered_words_input"])

    # Cross join with the full dataset and compute Jaccard similarity for each row
    recs = (
        processed.crossJoin(input_df)
                 .withColumn("jaccard", jaccard_udf("filtered_words", "filtered_words_input"))
                 .filter(col("title") != actual_title)
                 .orderBy(col("jaccard").desc())
                 .select("title", col("jaccard").alias("Similarity_Score"))
                 .limit(num_recs)
    )

    # Show the top N recommendations
    print(f"Top {num_recs} Recommendations for '{actual_title}':")
    recs.show(truncate=False)



recommend("Kota Factory", 5)
```

```
Target Title: Kota Factory
Top 5 Recommendations for 'Kota Factory':
+---------------------------------+-------------------+
|title                            |Similarity_Score   |
+---------------------------------+-------------------+
|Before 30                        |0.1891891891891892 |
|College Romance                  |0.17647058823529413|
|Lovestruck in the City           |0.16666666666666666|
|Little Things                    |0.16666666666666666|
|O-Negative, Love Can’t Be Designed|0.16666666666666666|
+---------------------------------+-------------------+
```