Machine Learning - IV

JHANVI PAREKH

60009210033
CSE (DS)

Experiment 9

Link: https://colab.research.google.com/drive/1ci-e-5-bapj2xh8TZ76py8ith1v2iwfA?usp=sharing

```
!pip install pyspark
```

```
Requirement already satisfied: pyspark in /usr/local/lib/python3.10/dist-packages (3.5.3)
Requirement already satisfied: py4j==0.10.9.7 in /usr/local/lib/python3.10/dist-packages (from pyspark) (0.10.9.7)
```

```
import zipfile
import os
from pyspark.sql import SparkSession


# Unzip the dataset
with zipfile.ZipFile('ml-latest-small.zip', 'r') as zip_ref:
    zip_ref.extractall('ml-latest-small')


# Check the contents of the extracted folder
print("Contents of the extracted folder:")
print(os.listdir('ml-latest-small'))
```

```
Contents of the extracted folder:
['ml-latest-small']
```

```
# Initialize a Spark session
spark = SparkSession.builder \
    .appName("MovieLens Analysis") \
    .getOrCreate()


# Check the contents of the extracted 'ml-latest-small' folder
print("Contents of the 'ml-latest-small' folder:")
print(os.listdir('ml-latest-small/ml-latest-small'))
```

```
Contents of the 'ml-latest-small' folder:
['README.txt', 'links.csv', 'tags.csv', 'movies.csv', 'ratings.csv']
```

```
import zipfile
import os

# Unzip the dataset
with zipfile.ZipFile('ml-latest-small.zip', 'r') as zip_ref:
    zip_ref.extractall('ml-latest-small')

# Check the contents of the extracted folder
print("Contents of the 'ml-latest-small' folder:")
print(os.listdir('ml-latest-small'))

# Check contents of the inner directory if it exists
if 'ml-latest-small' in os.listdir('ml-latest-small'):
    print("Contents of the inner 'ml-latest-small' folder:")
    print(os.listdir('ml-latest-small/ml-latest-small'))
```

```
Contents of the 'ml-latest-small' folder:
['ml-latest-small']
Contents of the inner 'ml-latest-small' folder:
['README.txt', 'links.csv', 'tags.csv', 'movies.csv', 'ratings.csv']
```

```
# Load the ratings dataset with the correct path
ratings_df = spark.read.csv("ml-latest-small/ml-latest-small/ratings.csv", header=True, inferSchema=True)

# Show the dataset structure
ratings_df.show()
```

```
+------+-------+------+---------+
|userId|movieId|rating|timestamp|
+------+-------+------+---------+
|     1|      1|   4.0|964982703|
|     1|      3|   4.0|964981247|
|     1|      6|   4.0|964982224|
|     1|     47|   5.0|964983815|
|     1|     50|   5.0|964982931|
|     1|     70|   3.0|964982400|
|     1|    101|   5.0|964980868|
```

```
|     1|    110|   4.0|964982176|
|     1|    151|   5.0|964984041|
|     1|    157|   5.0|964984100|
|     1|    163|   5.0|964983650|
|     1|    216|   5.0|964981208|
|     1|    223|   3.0|964980985|
|     1|    231|   5.0|964981179|
|     1|    235|   4.0|964980908|
|     1|    260|   5.0|964981680|
|     1|    296|   3.0|964982967|
|     1|    316|   3.0|964982310|
|     1|    333|   5.0|964981179|
|     1|    349|   4.0|964982563|
+------+-------+------+---------+
only showing top 20 rows
```

```
import os

# Check the contents of the extracted folder
print(os.listdir('ml-latest-small'))
```

```
['ml-latest-small']
```

```
import zipfile
import os
from pyspark.sql import SparkSession

# Unzip the dataset
with zipfile.ZipFile('ml-latest-small.zip', 'r') as zip_ref:
    zip_ref.extractall('ml-latest-small')

# Check the contents of the extracted folder
print("Contents of the 'ml-latest-small' folder:")
print(os.listdir('ml-latest-small'))

# Check contents of the inner directory
print("Contents of the inner 'ml-latest-small' folder:")
print(os.listdir('ml-latest-small/ml-latest-small'))

# Initialize a Spark session
spark = SparkSession.builder \
    .appName("MovieLens Analysis") \
    .getOrCreate()

# Load the ratings dataset with the correct path
ratings_df = spark.read.csv("ml-latest-small/ml-latest-small/ratings.csv", header=True, inferSchema=True)

# Show the dataset structure
ratings_df.show()

# You can proceed with the analysis from here...
```

```
Contents of the 'ml-latest-small' folder:
['ml-latest-small']
Contents of the inner 'ml-latest-small' folder:
['README.txt', 'links.csv', 'tags.csv', 'movies.csv', 'ratings.csv']
+------+-------+------+---------+
|userId|movieId|rating|timestamp|
+------+-------+------+---------+
|     1|      1|   4.0|964982703|
|     1|      3|   4.0|964981247|
|     1|      6|   4.0|964982224|
|     1|     47|   5.0|964983815|
|     1|     50|   5.0|964982931|
|     1|     70|   3.0|964982400|
|     1|    101|   5.0|964980868|
|     1|    110|   4.0|964982176|
|     1|    151|   5.0|964984041|
|     1|    157|   5.0|964984100|
|     1|    163|   5.0|964983650|
|     1|    216|   5.0|964981208|
|     1|    223|   3.0|964980985|
|     1|    231|   5.0|964981179|
|     1|    235|   4.0|964980908|
|     1|    260|   5.0|964981680|
|     1|    296|   3.0|964982967|
```

```
|     1|    316|   3.0|964982310|
|     1|    333|   5.0|964981179|
|     1|    349|   4.0|964982563|
+------+-------+------+---------+
only showing top 20 rows
```

```
# List all the movies and the number of ratings
movies_count = ratings_df.groupBy("movieId").count()
movies_count.show()
```

```
+-------+-----+
|movieId|count|
+-------+-----+
|   1580|  165|
|   2366|   25|
|   3175|   75|
|   1088|   42|
|  32460|    4|
|  44022|   23|
|  96488|    4|
|   1238|    9|
|   1342|   11|
|   1591|   26|
|   1645|   51|
|   4519|    9|
|   2142|   10|
|    471|   40|
|   3997|   12|
|    833|    6|
|   3918|    9|
|   7982|    4|
|   1959|   15|
|  68135|   10|
+-------+-----+
only showing top 20 rows
```

```
# List all the Movie IDs which have been rated
rated_movies = ratings_df.select("movieId").distinct()
rated_movies.show()
```

```
+-------+
|movieId|
+-------+
|   1580|
|   2366|
|   3175|
|   1088|
|  32460|
|  44022|
|  96488|
|   1238|
|   1342|
|   1591|
|   1645|
|   4519|
|   2142|
|    471|
|   3997|
|    833|
|   3918|
|   7982|
|   1959|
|  68135|
+-------+
only showing top 20 rows
```

```
# List all the Users who have rated the movies
rated_users = ratings_df.select("userId").distinct()
rated_users.show()
```

```
+------+
|userId|
+------+
|   148|
```

```
|   463|
|   471|
|   496|
|   243|
|   392|
|   540|
|    31|
|   516|
|    85|
|   137|
|   251|
|   451|
|   580|
|    65|
|   458|
|    53|
|   255|
|   481|
|   588|
+------+
only showing top 20 rows
```

```
# List of all the Users with max, min, average ratings
user_ratings = ratings_df.groupBy("userId") \
    .agg(
        {"rating": "max", "rating": "min", "rating": "avg"}
    ).withColumnRenamed("max(rating)", "max_rating") \
     .withColumnRenamed("min(rating)", "min_rating") \
     .withColumnRenamed("avg(rating)", "avg_rating")

user_ratings.show()
```

```
+------+------------------+
|userId|        avg_rating|
+------+------------------+
|   148|3.7395833333333335|
|   463| 3.787878787878788|
|   471|             3.875|
|   496| 3.413793103448276|
|   243| 4.138888888888889|
|   392|               3.2|
|   540|               4.0|
|    31|              3.92|
|   516|3.6923076923076925|
|    85|3.7058823529411766|
|   137| 3.978723404255319|
|   251| 4.869565217391305|
|   451|3.7941176470588234|
|   580| 3.529816513761468|
|    65| 4.029411764705882|
|   458|4.1525423728813555|
|    53|               5.0|
|   255|2.5681818181818183|
|   481| 2.806451612903226|
|   588|              3.25|
+------+------------------+
only showing top 20 rows
```

```
Start coding or generate with AI.
```