

TUGAS UJIAN TENGAH SEMESTER
BIG DATA DAN ANALITIK



Disusun oleh:

Handoko Wisnu Murti

19/444054/TK/49250

DEPARTEMEN TEKNIK ELEKTRO DAN TEKNOLOGI INFORMASI

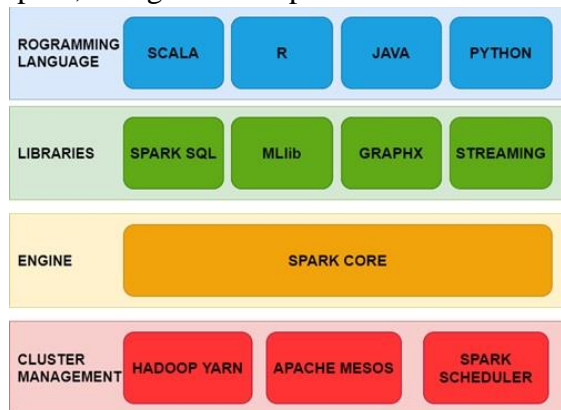
FAKULTAS TEKNIK

UNIVERSITAS GADJAH MADA

2022

A. Pengantar

Volume data yang dihasilkan melalui teknologi belakangan ini menjadi sangat besar. Penggunaan internet yang sangat masif menghasilkan data yang besar pula. Bahkan tidak sedikit sekumpulan data yang begitu besar atau kompleks tersebut tidak bisa ditangani lagi dengan sistem teknologi komputer konvensional. Inilah yang disebut sebagai Big Data (Hurwiz, et al. 2013). Diperlukan suatu teknologi untuk bisa menampung pemrosesan dan penyimpanan semua data yang dihasilkan dari teknologi tersebut. Saat ini sudah terdapat berbagai pilihan untuk melakukan pemrosesan *big data*, salah satunya adalah Apache Spark. Apache Spark adalah framework open-source yang mampu melakukan pendistribusian berskala besar yang diatur ke dalam banyak kluster komputer dengan memanfaatkan program yang sederhana. Apache Spark dapat digunakan pengguna untuk membaca, mentransformasi, dan mengagregasi data. Bahkan Apache Spark dapat melatih dan membuat model statistika dengan mudah. Apache Spark dapat dijalankan dengan beberapa Bahasa pemrograman, yaitu Scala, Java, Python, dan R. Terdapat beberapa komponen yang dimiliki pada Apache Spark, sebagaimana seperti visualisasi berikut.



Gambar 1: Komponen Apache Spark

(Sumber: <https://socs.binus.ac.id/2021/10/07/apache-spark-untuk-pemrosesan-big-data/>)

1. Spark Core

Merupakan mesin utama yang melakukan pemrosesan data secara parallel dan terdistribusi dengan skala yang besar. Komponen ini berisi berbagai fungsionalitas utama yang berjalan pada spark seperti penjadwalan tugas, manajemen memori, interaksi dengan sistem penyimpanan, serta penyimpanan konfigurasi *library* tambahan.

2. Library pada Spark

Terdapat berbagai library yang dapat digunakan pada Spark, diantaranya adalah Spark SQL, MLlib, GraphX, Streaming, dll. Spark SQL merupakan *library* yang mampu mengintegrasikan pemrosesan data relasional. Spark SQL mendukung pengolahan data berbasis kueri yang dapat dijalankan melalui Bahasa SQL ataupun Bahasa kueri Hive. MLlib merupakan *library* yang berisi berbagai jenis model *machine learning*, baik model *supervised learning*, *unsupervised learning*, regresi maupun model klasifikasi. GraphX adalah *library* yang dapat menjalankan komputasi grafik secara paralel. Grafik

tersebut memiliki arah pada setiap vertex dan edge yang memiliki property seperti nama, jarak, arah, dan informasi lain yang tersimpan dalam format RDD (*Resilient Distributed Dataset*). Sementara Streaming merupakan komponen yang dapat digunakan untuk memproses data secara *real time streaming*.

3. Spark Cluster Management

Terdapat beberapa *Cluster Management* yang terdapat pada Apache Spark, antara lain Apache Mesos, Apache Hadoop YARN, dan Spark Scheduler. Apache Mesos merupakan *cluster manager* yang juga dapat menjalankan aplikasi Hadoop. Apache Hadoop YARN berfungsi sebagai *resource manager*. Sedangkan Spark Scheduler berfungsi sebagai pengatur jadwal tugas suatu komputasi.

Dalam tugas ini, terdapat penjelasan pengimplementasian pengolahan Big Data menggunakan Apache Spark yang berjalan dengan Bahasa pemrograman python. Apache Spark yang berjalan dengan Bahasa pemrograman Python disebut dengan istilah Pyspark

B. Identifikasi Permasalahan

Contoh use case yang dapat diimplementasikan menggunakan Pyspark adalah penerapan konsep sistem rekomendasi. Sistem rekomendasi berguna untuk memprediksi jenis produk yang kemungkinan disukai oleh pengguna. Produk-produk tersebutlah yang akan direkomendasikan ke pengguna. Sebagai contoh, Youtube yang mampu memberikan rekomendasi video kepada pengguna berdasarkan video yang pernah ditonton sebelumnya dan Spotify yang mampu memberikan rekomendasi lagu berdasarkan genre ataupun jenis lagu yang didengarkan pengguna.

Pada projek ini, *use case* rekomendasi yang dibuat adalah sistem yang memberikan rekomendasi film kepada pengguna. Sistem rekomendasi ini menggunakan model *Alternating Least Squares* (ALS). Dataset yang digunakan berasal dari website <https://movielens.org> yang dikumpulkan oleh peneliti dari GroupLens. Dataset tersebut dikumpulkan dari berbagai periode waktu tertentu. terdapat dua dataset yang digunakan, yaitu dataset movies dan ratings. Dataset movies memiliki 3 atribut, yaitu movieID, title, dan genres. Datasets movies memiliki 9742 baris. Sedangkan dataset ratings memiliki 4 atribut, yaitu userId, movieId, rating, dan timestamp. Dataset rating memiliki 100.836 baris.

movieId	title	genres	userId	movieId	rating	timestamp
1	Toy Story (1995)	Adventure Animati...	1	1	4.0	964982703
2	Jumanji (1995)	Adventure Childre...	1	3	4.0	964981247
3	Grumpier Old Men ...	Comedy Romance	1	6	4.0	964982224
4	Waiting to Exhale...	Comedy Drama Romance	1	47	5.0	964983815
5	Father of the Bri...	Comedy	1	50	5.0	964982931
6	Heat (1995)	Action Crime Thri...	1	70	3.0	964982400
7	Sabrina (1995)	Comedy Romance	1	101	5.0	964980868
8	Tom and Huck (1995)	Adventure Children	1	110	4.0	964982176
9	Sudden Death (1995)	Action	1	151	5.0	964984041
10	GoldenEye (1995)	Action Adventure ...	1	157	5.0	964984100

Gambar 2: dataset movie (kiri) dan dataset rating (kanan)

Hasil akhir yang diinginkan ada proyek ini adalah memberikan rekomendasi film kepada pengguna berdasarkan model ALS yang telah dibuat.

C. Implementasi dan Pemecahan Permasalahan

a. Instalasi pyspark

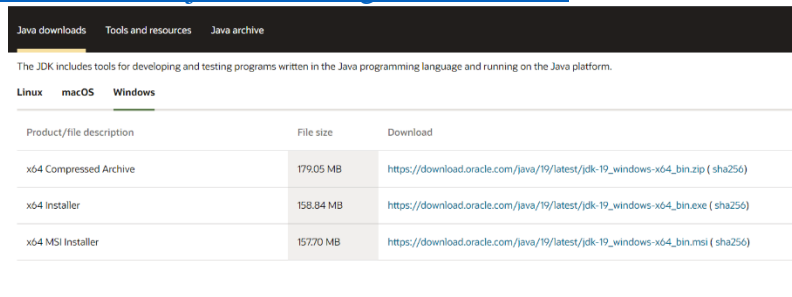
Instalasi Pyspark dilakukan di penyimpanan lokal laptop yang memiliki OS Windows 10. Untuk bisa menjalankan Pyspark pada penyimpanan lokal laptop, perlu melakukan beberapa tahapan sebagai berikut:

i. Install Python atau Anaconda

Python diunduh melalui python.org, sedangkan Anaconda diunduh melalui anaconda.com. Kemudian dilakukan instalasi program tersebut.

ii. Instal Java 8

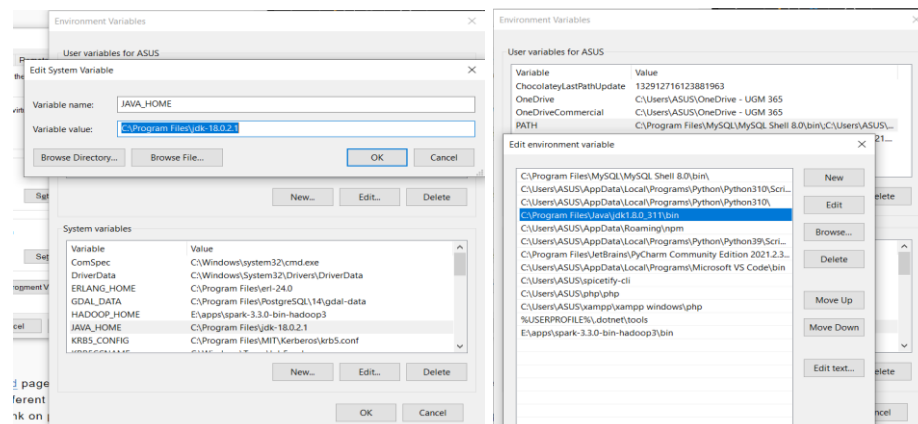
untuk menjalankan PySpark, perlu melakukan instalasi Java 8 ataupun versi yang lebih terbaru. Java diunduh melalui www.oracle.com/java/technologies/downloads/.



Java downloads Tools and resources Java archive		
The JDK includes tools for developing and testing programs written in the Java programming language and running on the Java platform.		
Linux macOS Windows		
Product/file description	File size	Download
x64 Compressed Archive	179.05 MB	https://download.oracle.com/java/19/latest/jdk-19_windows-x64_bin.zip (sha256)
x64 Installer	158.84 MB	https://download.oracle.com/java/19/latest/jdk-19_windows-x64_bin.exe (sha256)
x64 MSI Installer	157.70 MB	https://download.oracle.com/java/19/latest/jdk-19_windows-x64_bin.msi (sha256)

Gambar 3: Tampilan website untuk mengunduh java pada Oracle

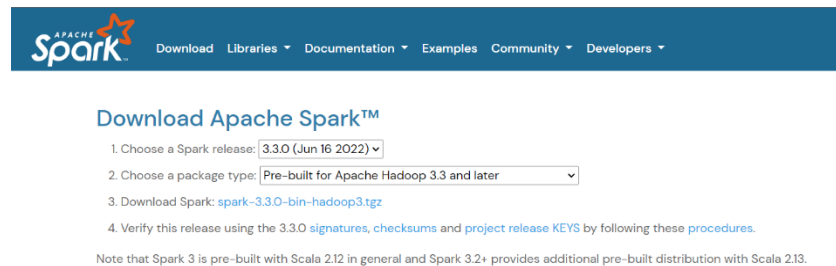
Agar java bisa berjalan, perlu mengonfigurasi *environment variable* pada Laptop. Terdapat dua variabel yang perlu dibuat, yaitu JAVA_HOME dan PATH.



Gambar 4: Konfigurasi pada environment variable

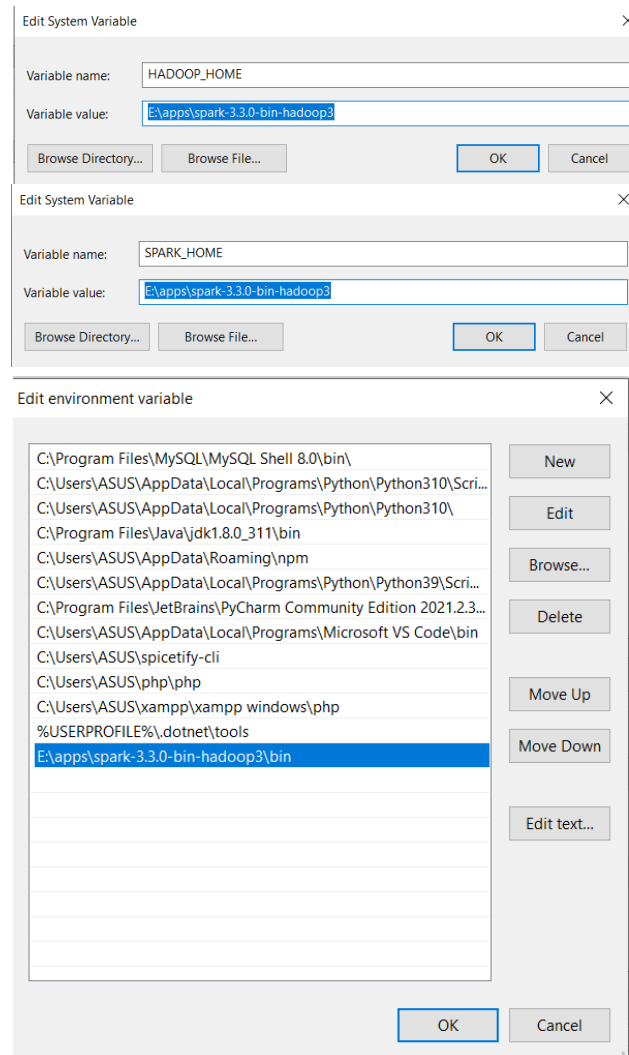
iii. Mengunduh apache spark

Apache Spark diunduh melalui spark.apache.org/downloads.html. Apache Spark yang diunduh memiliki versi 3.3.0.



Gambar 5: Tampilan Website untuk instalasi Apache Spark

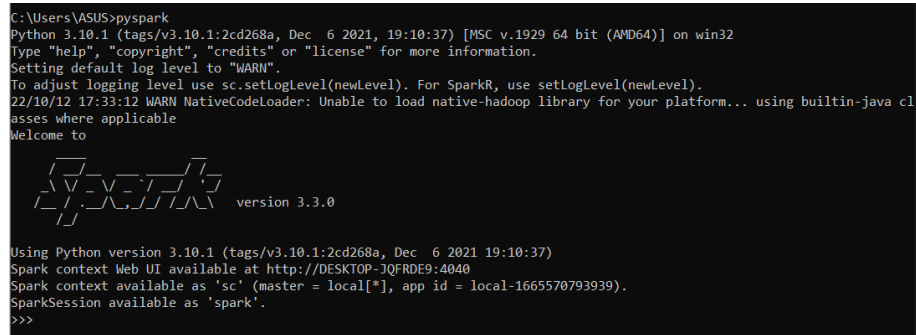
Berkas yang terunduh memiliki ekstensi .tar sehingga perlu dilakukan ekstraksi terlebih dahulu. Kemudian perlu melakukan konfigurasi terhadap *environment variabel* agar spark bisa dijalankan.



Gambar 6: Instalasi Apache Spark pada Environment Variabel

iv. Pyspark berhasil diinstal di local

Pyspark berhasil terinstal ditandai dengan *Command Prompt* yang mampu menjalankan perintah 'pyspark' menjalankan perintah 'pyspark' pada *Command Prompt*.



```
C:\Users\ASUS>pyspark
Python 3.10.1 (tags/v3.10.1:2cd268a, Dec 6 2021, 19:10:37) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
22/10/12 17:33:12 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java cl
asses where applicable
Welcome to

  ____      __
 /    |    /  |
|  __ |    /   | |
|  _ \|   /    |
| |_) |  /     |
|  __ < /      |
|_|  \_|/       |
           |_____|

version 3.3.0

Using Python version 3.10.1 (tags/v3.10.1:2cd268a, Dec 6 2021 19:10:37)
Spark context Web UI available at http://DESKTOP-1QFRDE9:4040
Spark context available as 'sc' (master = local[*], app id = local-1665570793939).
SparkSession available as 'spark'.
>>>
```

Gambar 7: Tampilan command prompt jika pyspark berhasil terinstal

b. Menjalankan program di Jupyter Notebook

Studi kasus yang dijelaskan pada bagian identifikasi permasalahan dipecahkan melalui program yang dijalankan di Jupyter Notebook. Jupyter notebook adalah platform aplikasi web yang dapat digunakan untuk membuat ataupun membagikan dokumen komputasional.

i. Mengonfigurasi pyspark pada jupyter notebook

Untuk menguji apakah jupyter notebook dapat menjalankan pyspark, perlu memanggil library findspark yang berfungsi untuk mendeteksi pyspark pada penyimpanan lokal laptop.

```
In [1]: import findspark
        findspark.init()
        findspark.find()

Out[1]: 'E:\\apps\\spark-3.3.0-bin-hadoop3'
```

Gambar 8: kode untuk mencari file spark pada penyimpanan lokal

Metode .init() memiliki fungsi untuk menginisiasi library findspark. Sementara metode .find() berfungsi untuk mencari file pyspark di penyimpanan lokal. Jika telah menghasilkan output berupa jalur penyimpanan pyspark maka pyspark dapat dijalankan pada jupyter notebook.

ii. Import library

Terdapat 3 library yang diimport pada proyek ini, yaitu col, explode, , dan SparkSession. Library col berfungsi untuk mengembalikan nilai berbasis kolom. Ini akan berguna jika ingin memanipulasi kolom. Library explode berfungsi untuk membuat array ataupun memetakan kolom ke baris. Library SparkSession bisa digunakan untuk membuat dataframe, meregistrasi

dataframe sebagai suatu table, mengeksekusi SQL pada table, dan menyembunyikan table.

```
In [2]: from pyspark.sql.functions import col, explode
        from pyspark.sql import SparkSession
```

Gambar 9: Import library pyspark.sql

iii. Memulai sesi spark yang baru

Sesi spark yang baru dibangun menggunakan library SparkSession. Sesi spark ini diberikan nama 'Recommendation'. Sesi ini disimpan pada variabel yang bernama 'spark'.

Memulai sesi Spark baru

```
In [3]: spark = SparkSession.builder.appName('Recommendation').getOrCreate()
```

Gambar 10: Memulai sesi spark yang baru

iv. Load dataset ke spark

Dataset memiliki ekstensi .csv dan disimpan pada penyimpanan lokal. Dataset diakses dengan kode 'spark.read.csv("file:lokasi_dataset")'. Terdapat dua dataset yang digunakan, yaitu ratings dan movies.

Load Data .csv into spark

```
In [4]: movies = spark.read.csv("file:///E:/t
        ratings = spark.read.csv("file:///E:/t
```

Gambar 11: Membaca dataset dari penyimpanan lokal

v. Eksplorasi data

1. Menampilkan dataset

Dataset ditampilkan dengan menggunakan metode .show(n). n bisa diisi dengan parameter angka yang bermakna jumlah baris yang ingin ditampilkan. Jika tidak diisi parameter, secara bawaan akan menampilkan 20 baris.

```
In [5]: movies.show(10)
```

movieId	title	genres
1	Toy Story (1995)	Adventure Animati...
2	Jumanji (1995)	Adventure Childre...
3	Grumpier Old Men ...	Comedy Romance
4	Waiting to Exhale...	Comedy Drama Romance
5	Father of the Bri...	Comedy
6	Heat (1995)	Action Crime Thri...
7	Sabrina (1995)	Comedy Romance
8	Tom and Huck (1995)	Adventure Children
9	Sudden Death (1995)	Action
10	GoldenEye (1995)	Action Adventure ...

only showing top 10 rows

```
In [8]: ratings.show(10)
```

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

only showing top 10 rows

Gambar 12: Menampilkan peninjauan pada dataset

2. Menghitung jumlah baris pada dataset
Jumlah baris pada dataframe diketahui dengan menggunakan metode `.count()`. Dataset movies memiliki 9742 baris, sedangkan rating memiliki 100.836 baris.

```
In [6]: movies.count()      In [9]: ratings.count()
Out[6]: 9742                Out[9]: 100836
```

Gambar 13: Jumlah baris pada setiap dataset

3. Menampilkan skema pada dataset
.KSkema ditampilkan dengan menggunakan metode `.printSchema()`. Skema ditampilkan dengan format *tree*. Luaran dari metode ini berupa daftar nama kolom pada dataframe

```
In [7]: movies.printSchema()      In [10]: ratings.printSchema()
root                                root
 |-- movieId: string (nullable = true)  |-- userId: string (nullable = true)
 |-- title: string (nullable = true)   |-- movieId: string (nullable = true)
 |-- genres: string (nullable = true)  |-- rating: string (nullable = true)
                                     |-- timestamp: string (nullable = true)
```

Gambar 14: Skema masing-masing dataset

4. Menghilangkan salah satu kolom pada dataset rating
Pada dataset rating, dilakukan pengurangan kolom timestamp. Ini dilakukan karena kolom timestamp tidak terlalu dibutuhkan pada proyek ini. pengurangan kolom ini dilakukan dengan menggunakan metode `.drop('NAMA_KOLOM')`.

```
In [11]: ratings = ratings.\
withColumn('userId', col('userId').cast('integer')).\
withColumn('movieId', col('movieId').cast('integer')).\
withColumn('rating', col('rating').cast('float')).\
drop('timestamp')

ratings.show()

+-----+-----+-----+
|userId|movieId|rating|
+-----+-----+-----+
| 1| 1| 4.0|
| 1| 3| 4.0|
| 1| 6| 4.0|
| 1| 47| 5.0|
| 1| 50| 5.0|
| 1| 70| 3.0|
| 1| 101| 5.0|
| 1| 110| 4.0|
| 1| 151| 5.0|
| 1| 157| 5.0|
| 1| 163| 5.0|
| 1| 216| 5.0|
| 1| 223| 3.0|
| 1| 231| 5.0|
| 1| 235| 4.0|
| 1| 260| 5.0|
| 1| 296| 3.0|
| 1| 316| 3.0|
| 1| 333| 5.0|
| 1| 349| 4.0|
+-----+-----+-----+
only showing top 20 rows
```

Gambar 15: Pengurangan kolom yang tidak dibutuhkan

5. Mengalkulasikan banyaknya kemunculan data berdasarkan suatu atribut

Proses ini dilakukan untuk mengetahui seberapa banyak seorang pengguna muncul pada data ratings. Terdapat beberapa metode yang digunakan, yaitu `.groupBy("Nama_kolom")`, `count()`, dan `orderBy('kolom')`. Parameter pada metode `groupBy` berguna untuk mengelompokkan suatu data berdasarkan atribut tertentu. Metode `.count()` digunakan untuk menghitung seberapa banyak munculnya data pengguna. Sedangkan metode `.orderBy()` digunakan untuk pengurutan data. Pengurutan dilakukan berdasarkan kolom `count` yang diurutkan dari nilai terbanyak.

```
In [14]: userId_ratings = ratings.groupBy("userId").count().orderBy('count', ascending=False)
userId_ratings.show()
```

userId	count
414	2698
599	2478
474	2108
448	1864
274	1346
610	1302
68	1260
380	1218
606	1115
288	1055
249	1046
387	1027
182	977
307	975
603	943
298	939
177	904
318	879
232	862
480	836

only showing top 20 rows

```
In [15]: movieId_ratings = ratings.groupBy("movieId").count().orderBy('count', ascending=False)
movieId_ratings.show()
```

movieId	count
356	329
318	317
296	307
593	279
2571	278
260	251
480	238
110	237
589	224
527	220
2959	218
1	215
1196	211
2858	204
50	204
47	203
780	202
150	201
1198	200
4993	198

only showing top 20 rows

Gambar 16: Jumlah kemunculan data `userId` dan `movieId`.

vi. Membuat model Alternate Least Square (ALS)

Pyspark mendukung penggunaan metode *machine learning*. Pyspark sudah memiliki berbagai *library* yang bisa digunakan untuk mengkomputasikan *machine learning*. Dalam proyek ini, metode *machine learning* yang digunakan adalah Alternate Least Square. ALS merupakan model *machine learning* yang berbasis faktorisasi matriks. Ini dipilih karena ALS mampu menangani dataset yang sangat *sparse*. Dataset yang *sparse* merupakan dataset yang memiliki nilai

null yang sangat banyak. Dataset tersebut bukan berisi nilai yang kosong melainkan memiliki nilai null ataupun NaN.

userid	4	5	10	14	15	18	19	26	31	34	...	283199	283204	283206	283208	283210	283215	283219	283222	283224
movieid																				
1	4.0	NaN	5.0	4.5	4.0	NaN	NaN	NaN	5.0	NaN	...	5.0	NaN	NaN	4.5	NaN	4.0	4.0	NaN	NaN
2	4.0	NaN	NaN	4.0	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	NaN	NaN	NaN	NaN	NaN	NaN	4.0	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	5.0	NaN	NaN	4.0
4	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
5	2.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	5.0	NaN	NaN	NaN
6	4.5	NaN	NaN	NaN	NaN	3.0	4.0	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
7	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	5.0	NaN	NaN	NaN
8	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
9	NaN	NaN	NaN	NaN	NaN	NaN	4.0	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
10	4.0	NaN	NaN	NaN	NaN	3.0	4.0	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	3.0
11	3.5	NaN	NaN	NaN	NaN	NaN	4.0	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	3.0	NaN	NaN	NaN
12	NaN	NaN	NaN	NaN	NaN	NaN	3.0	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
13	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
14	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	3.0
15	NaN	NaN	NaN	NaN	NaN	NaN	3.0	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Gambar 17: Ilustrasi dataset yang sparse. Sumber:

<https://towardsdatascience.com/prototyping-a-recommender-system-step-by-step-part-2-alternating-least-square-als-matrix-4a76c58714a1>

Dataset rating memiliki sparsity yang sangat tinggi. Jika dikalkulasikan, dataset rating memiliki persentase sparsity sebesar 98.30%. Ini berarti dataset rating memiliki nilai null yang sangat banyak

```
In [12]: # fungsi untuk mengalkulasikan sparsity: (1 - (Totalnonzero/total_elements))*100

def calc_sparsity(ratings):
    # menghitung jumlah data pada dataset rating
    nonzero=ratings.select("rating").count()

    # menghitung total elemen pada userID dan movieId
    num_user=ratings.select("userId").distinct().count()
    num_movies=ratings.select("movieId").distinct().count()
    total_element=num_user*num_movies

    #menghitung sparsity pada dataset
    sparsity=(1.0-(nonzero)/total_element)*100
    print("Dataframe rating memiliki persentase sparsity sebesar", "%.2f" %sparsity)

In [13]: calc_sparsity(ratings)

Dataframe rating memiliki persentase sparsity sebesar 98.30
```

Gambar 18: Nilai sparsity dataset ratings.

1. Import library

Terdapat 4 library yang diimpor, yaitu RegressionEvaluator, ALS, serta ParamGridBuilder dan CrossValidator. RegressionEvaluator digunakan untuk mengevaluasi model. ALS merupakan model *machine learning* yang akan digunakan. Sementara ParamGridBuilder dan CrossValidator digunakan untuk *tuning* model *machine learning*.

```
In [16]: from pyspark.ml.evaluation import RegressionEvaluator
from pyspark.ml.recommendation import ALS
from pyspark.ml.tuning import ParamGridBuilder, CrossValidator
```

Gambar 19: Import library pyspark untuk machine learning

2. Memisahkan data untuk training dan testing

Pyspark mendukung metode pemisahan data untuk training dan testing. Ini bisa dilakukan dengan menggunakan metode `.randomSplit()`. Pada proyek ini, data untuk training menggunakan 80% dari total dataset, sementara data untuk testing menggunakan 20% sisanya.

```
In [17]: # Membuat data untuk train dan test
(train, test) = ratings.randomSplit([0.8, 0.2], seed = 1234)
```

Gambar 20: Pemisahan data untuk training dan testing

3. Membuat model ALS

Model ALS disimpan dengan variabel `als`. Pemanggilan model ini disertai dengan beberapa parameter untuk menentukan nama kolom bagi user, item, dan rating. Kolom user diisi dengan `userId`, kolom item diisi dengan `movieId`, dan kolom rating diisi dengan `rating`.

```
# Membuat model ALS
als = ALS(userCol="userId", itemCol="movieId", ratingCol="rating", nonnegative = True, implicitPrefs = False, coldStartStrategy="
```

Gambar 21: Kode untuk membuat model Als

4. Menambahkan hyperparameter dan mendefinisikan evaluator

Hyperparameter digunakan untuk mengontrol proses *training* sehingga dapat membantu model untuk memaksimalkan performanya. Ini dilakukan sebelum proses *training* model dilakukan. Sementara evaluator digunakan untuk mengevaluasi hasil luaran. Metrics yang digunakan untuk evaluator adalah `rmse` (Root Mean Square Error). Metrik tersebut digunakan untuk menentukan akurasi prediksi yang dilakukan.

```
In [18]: # Add hyperparameters and their respective values to param_grid
param_grid = ParamGridBuilder() \
    .addGrid(als.rank, [10, 50, 100, 150]) \
    .addGrid(als.regParam, [.01, .05, .1, .15]) \
    .build()
#
    .addGrid(als.maxIter, [5, 50, 100, 200]) \

# Define evaluator as RMSE and print length of evaluator
evaluator = RegressionEvaluator(metricName="rmse", labelCol="rating", predictionCol="prediction") #rmse dipilih karena
print("Jumlah model yang diuji: ", len(param_grid))

Jumlah model yang diuji: 16
```

Gambar 22: Kode untuk membuat hyperparameter dan evaluator

5. Membuat pipeline untuk Cross Validation

Cross validation merupakan metode statistik yang digunakan untuk mengevaluasi kinerja suatu model *machine learning* dengan memisahkan data menjadi dua subset, yaitu data *training* dan data evaluasi.

```
In [19]: cv = CrossValidator(estimator=als, estimatorParamMaps=param_grid, evaluator=evaluator, numFolds=5)
```

Gambar 23: Kode untuk membuat Cross Validation

6. Menentukan model ALS yang terbaik

Setelah model dilakukan cross validation, perlu dipilih model yang terbaik. pemilihan model terbaik ini bisa dilakukan oleh kelas CrossValidator. Ini dapat dicapai dengan menambahkan .bestModel pada model yang telah dilatih dengan data *training*.

```
In [20]: #Fitting Cross validator ke dataset training
model = cv.fit(train)

#Mengekstraksi Model Terbaik
best_model = model.bestModel
```

Gambar 24: Kode untuk menentukan model terbaik

7. Mengalkulasikan akurasi model

Model yang telah dilakukan proses pelatihan, perlu diketahui tingkat akurasinya. Semakin tinggi tingkat akurasi, semakin baik model tersebut mampu melakukan prediksi. Menentukan tingkat akurasi dilakukan dengan menggunakan evaluator yang telah didefinisikan pada tahapan yang sebelumnya. Model yang dibuat pada proyek ini memiliki tingkat akurasi sekitar 86%

```
In [22]: test_predictions = best_model.transform(test)
RMSE = evaluator.evaluate(test_predictions)
print(RMSE)

0.8688284419825071
```

Gambar 25: Tingkat akurasi model

vii. Memberikan rekomendasi kepada semua pengguna

1. Mengimplementasikan model terbaik untuk merekomendasikan film

Sampai tahapan ini, model telah siap digunakan untuk memberikan rekomendasi kepada setiap pengguna. Tahapan selanjutnya adalah memberikan rekomendasi film kepada setiap pengguna. Pemberian rekomendasi ini dapat dilakukan dengan menggunakan metode .recommendForAllUsers('n'). Secara otomatis, setiap pengguna akan

```
In [29]: nrecommendations = best_model.recommendForAllUsers(15)
nrecommendations.limit(10).show()
```

```
+-----+-----+
|userId| recommendations|
+-----+-----+
| 1|[{3379, 5.7612557...}|
| 3|[{5746, 4.858056}...}|
| 5|[{3379, 4.555614}...}|
| 6|[{3925, 4.827505}...}|
| 12|[{45503, 5.696589...}|
| 13|[{3379, 5.04909}...}|
| 15|[{60943, 4.460198...}|
| 16|[{3379, 4.460198...}|
```

Gambar 26: Kode untuk mengimplementasikan model rekomendasi ke semua pengguna

mendapatkan rekomendasi film. Nilai n merupakan parameter angka yang dapat diisi sesuai dengan jumlah rekomendasi yang diinginkan.

2. Memisahkan array luaran dari model

Hasil rekomendasi berisi array yang memiliki dua nilai, yaitu movieId dan rating. Oleh karena itu, diperlukan pemisahan array sehingga kedua nilai tersebut dapat diletakkan pada dua kolom yang berbeda.

```
nrecommendations = nrecommendations\
    .withColumn("rec_exp", explode("recommendations"))\
    .select('userId', col("rec_exp.movieId"), col("rec_exp.rating"))

nrecommendations.limit(10).show()
```

userId	movieId	rating
1	3379	5.7612557
1	33649	5.6169276
1	5490	5.517828
1	171495	5.425299
1	3951	5.4084063
1	5328	5.4084063
1	5416	5.4084063
1	78836	5.3639174
1	8477	5.34965
1	5915	5.335429

Gambar 27: Kode untuk membuat 2 kolom baru atas pemisahan array

3. Hasil rekomendasi akhir bagi semua pengguna

Setelah melakukan semua tahapan diatas, semua pengguna telah mendapatkan rekomendasi film yang sesuai dengan profil pengguna.

```
In [34]: ratings.join(movies, on='movieId').filter('userId = 100').sort('rating', ascending=False).limit(10).show()
```

movieId	userId	rating	title	genres
1101	100	5.0	Top Gun (1986)	Action Romance
1958	100	5.0	Terms of Endearme...	Comedy Drama
2423	100	5.0	Christmas Vacatio...	Comedy
4041	100	5.0	Officer and a Gen...	Drama Romance
5620	100	5.0	Sweet Home Alabam...	Comedy Romance
368	100	4.5	Maverick (1994)	Adventure Comedy ...
934	100	4.5	Father of the Bri...	Comedy
539	100	4.5	Sleepless in Seat...	Comedy Drama Romance
16	100	4.5	Casino (1995)	Crime Drama
553	100	4.5	Tombstone (1993)	Action Drama Western

Gambar 28: Tabel rekomendasi akhir bagi semua pengguna

Daftar Pustaka

- Acharya, S. (2021, Mei 14). *What are RMSE and MAE?* Retrieved from Towards Data Science: <https://towardsdatascience.com/what-are-rmse-and-mae-e405ce230383>
- Bandi, R., Amudhavel, J., & Karthik, R. (2018). Machine Learning with PySpark - Review. *Indonesian Journal of Electrical Engineering and Computer Science*, 1.
- Drabas, T., & Lee, D. (2017). *Learning PySpark*. Packt.
- Liao, K. (2018, November 17). *Prototyping a Recommender System Step by Step Part 2: Alternating Least Square (ALS) Matrix Factorization in Collaborative Filtering*. Retrieved from Towards Data Science: <https://towardsdatascience.com/prototyping-a-recommender-system-step-by-step-part-2-alternating-least-square-als-matrix-4a76c58714a1>
- Nair, S. (2020, Agustus 10). *PySpark Collaborative Filtering with ALS*. Retrieved from Towards Data Science: <https://towardsdatascience.com/build-recommendation-system-with-pyspark-using-alternating-least-squares-als-matrix-factorisation-ebe1ad2e7679>
- Spark with Python (PySpark) Tutorial For Beginners*. (n.d.). Retrieved from SparkBy{Example}: <https://sparkbyexamples.com/pyspark-tutorial/>