Movie_recommendation

Cmd 1

```
print "sc type: ", type(sc)
print "Driver Program name: ", sc.appName
print "Spark version: ", sc.version
```

```
sc type:  <class '__main__.RemoteContext'>
Driver Program name:  Databricks Shell
Spark version:  2.1.1
```

Command took 0.16 seconds -- by meghana.rwgsql@gmail.com at 6/10/2017, 2:24:41 AM on Movies_Project

Cmd 2

```
moviesDF = spark.sql("select * from movies")
moviesDF.show(truncate = False)
```

▸ (1) Spark Jobs

```
+-------+-------------------------------------+--------------------------------------------
----------+
|movieId|title                                |genres
          |
+-------+-------------------------------------+--------------------------------------------
----------+
|1      |Toy Story (1995)                     |Adventure|Animation|Children|Come
dy|Fantasy|
|2      |Jumanji (1995)                       |Adventure|Children|Fantasy
          |
|3      |Grumpier Old Men (1995)              |Comedy|Romance
          |
|4      |Waiting to Exhale (1995)             |Comedy|Drama|Romance
          |
|5      |Father of the Bride Part II (1995)   |Comedy
          |
|6      |Heat (1995)                          |Action|Crime|Thriller
          |
|7      |Sabrina (1995)                       |Comedy|Romance
          |
|8      |Tom and Huck (1995)                  |Adventure|Children
          |
|9      |Sudden Death (1995)                  |Action
          |
|10     |GoldenEye (1995)                     |Action|Adventure|Thriller
          |
|11     |American President, The (1995)       |Comedy|Drama|Romance
          |
|12     |Dracula: Dead and Loving It (1995)   |Comedy|Horror
```

```
       |
|13     |Balto (1995)                        |Adventure|Animation|Children
       |
|14     |Nixon (1995)                        |Drama
       |
|15     |Cutthroat Island (1995)             |Action|Adventure|Romance
       |
|16     |Casino (1995)                       |Crime|Drama
       |
|17     |Sense and Sensibility (1995)        |Drama|Romance
       |
|18     |Four Rooms (1995)                   |Comedy
       |
|19     |Ace Ventura: When Nature Calls (1995)|Comedy
       |
|20     |Money Train (1995)                  |Action|Comedy|Crime|Drama|Thrille
r      |
+-------+-------------------------------------+--------------------------------
----------+
only showing top 20 rows
```

Command took 3.27 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 10:14:36 AM on movies_project

Cmd 3

```
spark.sql("SELECT count(*) FROM movies").show()
```

▸ (1) Spark Jobs

```
+--------+
|count(1)|
+--------+
|   27278|
+--------+
```

Command took 1.31 seconds -- by meghana.rwgsql@gmail.com at 8/30/2017, 9:09:07 PM on movies_project

Cmd 4

```
moviesDF.printSchema()
```

```
root
 |-- movieId: string (nullable = true)
 |-- title: string (nullable = true)
 |-- genres: string (nullable = true)
```

Command took 0.07 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 1:50:08 PM on movies_project

Cmd 5

```
ratingsDF = spark.sql("select * from ratings")
ratingsDF.show()
```

▸ (1) Spark Jobs

```
+------+-------+------+----------+
|userId|movieId|rating| timestamp|
+------+-------+------+----------+
|     1|      2|   3.5|1112486027|
|     1|     29|   3.5|1112484676|
|     1|     32|   3.5|1112484819|
|     1|     47|   3.5|1112484727|
|     1|     50|   3.5|1112484580|
|     1|    112|   3.5|1094785740|
|     1|    151|   4.0|1094785734|
|     1|    223|   4.0|1112485573|
|     1|    253|   4.0|1112484940|
|     1|    260|   4.0|1112484826|
|     1|    293|   4.0|1112484703|
|     1|    296|   4.0|1112484767|
|     1|    318|   4.0|1112484798|
|     1|    337|   3.5|1094785709|
|     1|    367|   3.5|1112485980|
|     1|    541|   4.0|1112484603|
|     1|    589|   3.5|1112485557|
|     1|    593|   3.5|1112484661|
|     1|    653|   3.0|1094785691|
|     1|    919|   3.5|1094785621|
+------+-------+------+----------+
only showing top 20 rows
```

```
Command took 0.43 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 10:17:28 AM on
movies_project
```

Cmd 6

```
spark.sql("SELECT count(*) FROM ratings").show()
```

▸ (1) Spark Jobs

```
+--------+
|count(1)|
+--------+
|20000263|
+--------+
```

```
Command took 8.88 seconds -- by meghana.rwgsql@gmail.com at 8/30/2017, 9:09:32 PM on
movies_project
```

Cmd 7

```
ratingsDF.printSchema()
```

```
root
 |-- userId: string (nullable = true)
 |-- movieId: string (nullable = true)
 |-- rating: string (nullable = true)
 |-- timestamp: string (nullable = true)
```

Command took 0.07 seconds -- by meghana.rwgsql@gmail.com at 8/30/2017, 12:25:52 AM on Movies_Project

Cmd 8

```
ratingsDF = ratingsDF.withColumn('rating', ratingsDF["rating"].cast("float"))
ratingsDF = ratingsDF.withColumn('userId', ratingsDF["userId"].cast("int"))
ratingsDF = ratingsDF.withColumn('movieId', ratingsDF["movieId"].cast("int"))
ratingsDF.show()
```

▸ (1) Spark Jobs

```
+------+-------+------+----------+
|userId|movieId|rating| timestamp|
+------+-------+------+----------+
|     1|      2|   3.5|1112486027|
|     1|     29|   3.5|1112484676|
|     1|     32|   3.5|1112484819|
|     1|     47|   3.5|1112484727|
|     1|     50|   3.5|1112484580|
|     1|    112|   3.5|1094785740|
|     1|    151|   4.0|1094785734|
|     1|    223|   4.0|1112485573|
|     1|    253|   4.0|1112484940|
|     1|    260|   4.0|1112484826|
|     1|    293|   4.0|1112484703|
|     1|    296|   4.0|1112484767|
|     1|    318|   4.0|1112484798|
|     1|    337|   3.5|1094785709|
|     1|    367|   3.5|1112485980|
|     1|    541|   4.0|1112484603|
|     1|    589|   3.5|1112485557|
|     1|    593|   3.5|1112484661|
|     1|    653|   3.0|1094785691|
|     1|    919|   3.5|1094785621|
+------+-------+------+----------+
only showing top 20 rows
```

```
Command took 0.53 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 10:17:37 AM on movies_project
```

Cmd 9

```python
from pyspark.sql.functions import from_unixtime
ratingsDF = ratingsDF.withColumn('timestamp', from_unixtime(ratingsDF.timestamp))

```

```
Command took 0.12 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 10:17:42 AM on movies_project
```

Cmd 10

```python
from pyspark.sql.types import TimestampType
ratingsDF = ratingsDF.withColumn('timestamp', ratingsDF.timestamp.cast(TimestampType()))
```

```
Command took 0.12 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 10:17:46 AM on movies_project
```

Cmd 11

```python
ratingsDF.printSchema()
```

```
root
 |-- userId: integer (nullable = true)
 |-- movieId: integer (nullable = true)
 |-- rating: float (nullable = true)
 |-- timestamp: timestamp (nullable = true)
```

```
Command took 0.07 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 10:17:51 AM on movies_project
```

Cmd 12

```python
moviesDF.describe().show()
```

(1) Spark Jobs

```
+-------+-----------------+--------------------+------------------+
|summary|          movieId|               title|            genres|
+-------+-----------------+--------------------+------------------+
|  count|            27278|               27278|             27278|
|   mean|59855.48057042305|                null|              null|
| stddev|44429.31469707313|                null|              null|
|    min|                1|"""Great Performa...|(no genres listed)|
|    max|            99999|         貞子3D (2012)|           Western|
+-------+-----------------+--------------------+------------------+
```

```
Command took 2.32 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 10:17:55 AM on movies_project
```

Cmd 13

```
#m_DF =
sqlContext.read.format("csv").load("/FileStore/tables/4riqczku1497073456815
/movies.csv")
#r_df =
sqlContext.read.format("csv").load("/FileStore/tables/f9oap0qt1497073661552
/ratings.csv")
```

▸ (2) Spark Jobs

```
Command took 0.93 seconds -- by meghana.rwgsql@gmail.com at 6/10/2017, 2:37:14 AM on
Movies_Project
```

Cmd 14

```
ratingsDF.describe().show()
```

▸ (1) Spark Jobs

```
+-------+-----------------+-----------------+------------------+
|summary|           userId|          movieId|            rating|
+-------+-----------------+-----------------+------------------+
|  count|         20000263|         20000263|          20000263|
|   mean|69045.87258292554|9041.567330339605|3.5255285642993797|
| stddev|40038.62665316182|19789.47744541297|  1.05198891929425|
|    min|                1|                1|               0.5|
|    max|           138493|           131262|               5.0|
+-------+-----------------+-----------------+------------------+
```

```
Command took 14.05 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 10:18:02 AM on
movies_project
```

Cmd 15

```
#To check if Dataframe contains None/blank values
print(ratingsDF[ratingsDF.rating == None].count())
print(ratingsDF[ratingsDF.userId == None].count())
print(ratingsDF[ratingsDF.movieId == None].count())
```

▸ (3) Spark Jobs

```
0
0
0
```

```
Command took 23.31 seconds -- by meghana.rwgsql@gmail.com at 7/20/2017, 5:23:05 PM on
Movies_Project
```

Cmd 16

```
#Counting number of users who gave highest ratings to the movie
num_users = spark.sql("SELECT count(userId) as no_of_user, ratings.movieId, movies.title, rating FROM ratings JOIN movies on movies.movieId = ratings.movieId  Group by ratings.movieId, movies.title, rating Order by rating DESC LIMIT 10")
num_users.show(truncate = False)

```

▸ (2) Spark Jobs

```
+----------+-------+---------------------------------------------------+------+
|no_of_user|movieId|title                                              |rating|
+----------+-------+---------------------------------------------------+------+
|83        |1005   |D3: The Mighty Ducks (1996)                        |5.0   |
|3         |7878   |Straight to Hell (1987)                            |5.0   |
|91        |688    |Operation Dumbo Drop (1995)                        |5.0   |
|2536      |1517   |Austin Powers: International Man of Mystery (1997)|5.0   |
|102       |6059   |Recruit, The (2003)                                |5.0   |
|231       |88810  |Help, The (2011)                                   |5.0   |
|210       |1888   |Hope Floats (1998)                                 |5.0   |
|8272      |924    |2001: A Space Odyssey (1968)                       |5.0   |
|13        |6454   |Music Box (1989)                                   |5.0   |
|94        |118    |If Lucy Fell (1996)                                |5.0   |
+----------+-------+---------------------------------------------------+------+
```

Command took 19.46 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 10:26:26 AM on movies_project

Cmd 17

```
#Retrieving the highest average ratings for each movies
Movies_avg_ratings = spark.sql("SELECT ratings.movieId, movies.title, avg(rating) as avg_rating FROM ratings JOIN movies on movies.movieId = ratings.movieId \
                         Group by ratings.movieId, movies.title Order by avg(rating) DESC LIMIT 20")
Movies_avg_ratings.show(truncate = False)
```

▸ (2) Spark Jobs

```
+-------+-----------------------------------------------------+----------+
|movieId|title                                                |avg_rating|
+-------+-----------------------------------------------------+----------+
|113790 |Peace, Propaganda & the Promised Land (2004)         |5.0       |
|111548 |Welcome to Australia (1999)                          |5.0       |
|129295 |A Gun for Jennifer (1997)                            |5.0       |
|116387 |Muddy River (1981)                                   |5.0       |
|94949  |Boy Meets Boy (2008)                                 |5.0       |
|94431  |Ella Lola, a la Trilby (1898)                        |5.0       |
|130996 |The Beautiful Story (1992)                             |5.0       |
|109253 |Argentina latente (2007)                               |5.0       |
|32230  |Snow Queen, The (Lumikuningatar) (1986)                |5.0       |
|103753 |Human Behavior Experiments, The (2006)                 |5.0       |
|89133  |Boys (Drenge) (1977)                                   |5.0       |
|100830 |Blue Swallow (Cheong yeon) (2005)                      |5.0       |
|106082 |Shock and Awe: The Story of Electricity (2011)         |5.0       |
|26718  |Life On A String (Bian chang Bian Zou) (1991)          |5.0       |
|94737  |Boys Diving, Honolulu (1901)                           |5.0       |
|131050 |Stargate SG-1 Children of the Gods - Final Cut (2009)|5.0       |
|103143 |Donos de Portugal (2012)                               |5.0       |
|121029 |No Distance Left to Run (2010)                         |5.0       |
|117061 |The Green (2011)                                       |5.0       |
|108527 |Catastroika (2012)                                     |5.0       |
+-------+-----------------------------------------------------+----------+
```

Command took 21.88 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 10:26:53 AM on movies_project

Cmd 18

```
#Counts number of movies rated by each user
num_Movies_rated_by_user = spark.sql("SELECT count(movieId) as no_of_movies, userId from ratings GROUP BY userId order by userId")
num_Movies_rated_by_user.show()


```

▸ (1) Spark Jobs

```
+------------+------+
|no_of_movies|userId|
+------------+------+
|         175|     1|
|          38|    10|
|          52|   100|
|          57|  1000|
|          73| 10000|
|         103|100000|
|          21|100001|
|          56|100002|
|          99|100003|
|         186|100004|
|          38|100005|
|         116|100006|
|          90|100007|
|         141|100008|
|         455|100009|
|         709| 10001|
|          35|100010|
|         152|100011|
|          20|100012|
|         226|100013|
+------------+------+
only showing top 20 rows
```

Command took 11.69 seconds -- by meghana.rwgsql@gmail.com at 8/30/2017, 4:15:03 PM on Movies_Recommendation

Cmd 19

```
#Counting distinct users and movies from ratings table
numUsers = spark.sql("SELECT count(Distinct userId) as no_of_Users from ratings")
numMovies = spark.sql("SELECT count(Distinct movieId) as no_of_Movies from ratings")
numUsers.show()
numMovies.show()
```

▸ (2) Spark Jobs

```
+-----------+
|no_of_Users|
+-----------+
|     138493|
+-----------+

+------------+
|no_of_Movies|
+------------+
|       26744|
+------------+
```

Command took 24.14 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 10:27:30 AM on movies_project

Cmd 20

```
ratings_count = ratingsDF.count()
movies_count = moviesDF.count()

print 'There are %s ratings and %s movies in the datasets' % (ratings_count, movies_count)
print 'Ratings:'
ratingsDF.show(3)
print 'Movies:'
moviesDF.show(3, truncate=False)

assert ratings_count == ratings_count
assert movies_count == movies_count
```

▸ (4) Spark Jobs

```
There are 20000263 ratings and 27278 movies in the datasets
Ratings:
+------+-------+------+-------------------+
|userId|movieId|rating|          timestamp|
+------+-------+------+-------------------+
|     1|      2|   3.5|2005-04-02 23:53:47|
|     1|     29|   3.5|2005-04-02 23:31:16|
|     1|     32|   3.5|2005-04-02 23:33:39|
+------+-------+------+-------------------+
only showing top 3 rows

Movies:
+-------+-----------------------+-------------------------------------------+
|movieId|title                  |genres                                     |
+-------+-----------------------+-------------------------------------------+
|1      |Toy Story (1995)       |Adventure|Animation|Children|Comedy|Fantasy|
|2      |Jumanji (1995)         |Adventure|Children|Fantasy                 |
|3      |Grumpier Old Men (1995)|Comedy|Romance                             |
+-------+-----------------------+-------------------------------------------+
only showing top 3 rows
```

Command took 8.93 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 10:30:18 AM on movies_project

Cmd 21

```
#MOVIES WITH HIGHEST AVERAGE RATINGS

from pyspark.sql import functions as F

# Rename column from movieId to ID
movies_DF = moviesDF.withColumnRenamed('movieId', 'ID')

# From ratingsDF, create a movie_ids_with_avg_ratings_df that combines the
two DataFrames
movie_ids_with_avg_ratings_df =
ratingsDF.groupBy('movieId').agg(F.count(ratingsDF.rating).alias("count"),
F.avg(ratingsDF.rating).alias("average"))
print 'movie_ids_with_avg_ratings_df:'
movie_ids_with_avg_ratings_df.show(3, truncate=False)

# Note: movie_names_df is a temporary variable, used only to separate the
steps necessary
# to create the movie_names_with_avg_ratings_df DataFrame.
#movie_names_df = movie_ids_with_avg_ratings_df.<FILL_IN>  # use
df.join(df2, df.name == df2.name)
movie_names_df = movie_ids_with_avg_ratings_df.join(movies_DF,
movie_ids_with_avg_ratings_df.movieId == movies_DF.ID)

#movie_names_with_avg_ratings_df = movie_names_df.<FILL_IN>
movie_names_with_avg_ratings_df = movie_names_df.select("average", "title",
"count", "movieId")

print 'movie_names_with_avg_ratings_df:'
movie_names_with_avg_ratings_df.show(3, truncate=False)
```

▸ (3) Spark Jobs

```
movie_ids_with_avg_ratings_df:
+-------+-----+------------------+
|movieId|count|average           |
+-------+-----+------------------+
|3997   |2047 |2.0703468490473864|
|1580   |35580|3.55831928049466  |
|3918   |1246 |2.918940609951846 |
+-------+-----+------------------+
only showing top 3 rows

movie_names_with_avg_ratings_df:
+------------------+---------------------------------+-----+-------+
|average           |title                            |count|movieId|
+------------------+---------------------------------+-----+-------+
|2.0703468490473864|Dungeons & Dragons (2000)        |2047 |3997   |
|3.55831928049466  |Men in Black (a.k.a. MIB) (1997)|35580|1580   |
|2.918940609951846 |Hellbound: Hellraiser II (1988) |1246 |3918   |
+------------------+--------------------------------+-----+-------+
only showing top 3 rows
```

Command took 26.81 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 10:30:32 AM on movies_project

Cmd 22

```
#We want to filter our movies with high ratings but greater than or equal to 500 reviews.
#movies_with_500_ratings_or_more = movie_names_with_avg_ratings_df.<FILL_IN>
movies_with_500_ratings_or_more = (movie_names_with_avg_ratings_df
                                   .where("count >= 500")
                                   .orderBy(movie_names_with_avg_ratings_df.average.desc()))

print 'Movies with highest ratings:'
movies_with_500_ratings_or_more.show(20, truncate=False)
```

▸ (2) Spark Jobs

```
Movies with highest ratings:
+------------------+------------------------------------------------------------------------------+-----+-------+
|average           |title                                                                         |count|movieId|
+------------------+------------------------------------------------------------------------------+-----+-------+
|4.446990499637029 |Shawshank Redemption, The (1994)                                              |63366|318    |
|4.364732196832306 |Godfather, The (1972)                                                         |41355|858    |
|4.334372207803259 |Usual Suspects, The (1995)                                                    |47006|50     |
|4.310175010988133 |Schindler's List (1993)                                                       |50054|527    |
|4.275640557704942 |Godfather: Part II, The (1974)                                                |27398|1221   |
|4.2741796572216   |Seven Samurai (Shichinin no samurai) (1954)                                   |11611|2019   |
|4.271333600779414 |Rear Window (1954)                                                            |17449|904    |
|4.263182346109176 |Band of Brothers (2001)                                                       |4305 |7502   |
|4.258326830670664 |Casablanca (1942)                                                             |24349|912    |
|4.256934865900383 |Sunset Blvd. (a.k.a. Sunset Boulevard) (1950)
               |6525 |922    |
|4.24807897901911  |One Flew Over the Cuckoo's Nest (1975)
               |29932|1193   |
|4.247286821705426 |Dr. Strangelove or: How I Learned to Stop Worrying and Love
 the Bomb (1964)|23220|750    |
|4.246001523229246 |Third Man, The (1949)
               |6565 |1212   |
|4.235410064157069 |City of God (Cidade de Deus) (2002)
               |12937|6016   |
|4.2347902097902095|Lives of Others, The (Das leben der Anderen) (2006)
               |5720 |44555  |
|4.233538107122288 |North by Northwest (1959)
               |15627|908    |
|4.2326233183856505|Paths of Glory (1957)
               |3568 |1178   |
|4.227123123722136 |Fight Club (1999)
               |40106|2959   |
|4.224281931146873 |Double Indemnity (1944)
               |4909 |3435   |
|4.224137931034483 |12 Angry Men (1957)
               |12934|1203   |
+------------------+--------------------------------------------------------------
---------------+-----+-------+
only showing top 20 rows
```

Command took 13.62 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 10:33:40 AM on movies_project

Cmd 23

```
#Find all the movies that have average ratings greater than 4.0.
#ratingsDF.groupBy("movieId").avg("rating").filter("avg(rating) >
4.0").show()
```

▸ (2) Spark Jobs

```
+-------+-----------------+
|movieId|      avg(rating)|
+-------+-----------------+
|  89056|            4.125|
|  78064|4.111111111111111|
|  81501|              4.5|
| 102119|              4.5|
|  97092|              4.5|
|    858|4.364732196832306|
|  48780|4.042195403318839|
| 118258|              4.5|
|   3226|4.666666666666667|
| 104317|               5.0|
|  97872|               4.5|
|   3000|4.096298619824341|
|   1303|4.040199611865816|
|  80906|4.005541346973572|
| 116183|               4.5|
|   3089|4.143596986817326|
| 104829|4.083333333333333|
|  56490|               4.1|
| 125599|               5.0|
|   1223|4.066765197275415|
+-------+-----------------+
only showing top 20 rows
```

Command took 10.31 seconds -- by meghana.rwgsql@gmail.com at 7/4/2017, 5:44:27 PM on Movies_Project

Cmd 24

```
#Find all the movies that have average ratings greater than 4.0 and have recieved more than 400 reviews

ratingsDF.groupBy("movieId")\
.agg({"rating":"avg", "*":"count"})\
.filter("avg(rating) > 4.0 AND count(1) > 400")\
.show()
```

▸ (3) Spark Jobs

```
+-------+-----------------+--------+
|movieId|      avg(rating)|count(1)|
+-------+-----------------+--------+
|    858|4.364732196832306|   41355|
|  48780|4.042195403318839|   11269|
|   3000|4.096298619824341|    9564|
|   1303|4.040199611865816|    3607|
|  80906|4.005541346973572|    1173|
|   3089|4.143596986817326|    3186|
|   1223|4.066765197275415|    7781|
|    898|4.171426401336777|    6583|
|   5995|4.053922967189729|   10515|
|    296|4.174231169217055|   67310|
|  68954|4.038266407599309|    9264|
|  86377| 4.08361391694725|     891|
|  58559|4.220129171151776|   20438|
|    593| 4.17705650958151|   63299|
|   1199|4.029590886293616|   13957|
|   1212|4.246001523229246|    6565|
|   7132|4.047279792746114|    1544|
|    950|   4.184187016081|    3358|
|   1198|4.219009123455364|   43295|
|   7587|4.065116279069767|     645|
+-------+-----------------+--------+
only showing top 20 rows
```

Command took 14.55 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 10:34:20 AM on movies_project

Cmd 25

```
#Find all the movies that have less than 10 reviews

ratingsDF.groupBy("movieId")\
.agg({"*":"count"})\
.filter("count(1) <10")\
.show()
```

▸ (1) Spark Jobs

```
+-------+--------+
|movieId|count(1)|
+-------+--------+
| 104064|       5|
|  96469|       7|
| 104508|       2|
|  46952|       9|
| 102798|       8|
|  80451|       9|
|   7754|       4|
| 119432|       4|
|  89056|       4|
|  89844|       6|
|  92182|       2|
|  78064|       9|
|  80033|       3|
|  83250|       9|
|  81349|       2|
| 103747|       3|
|  71995|       5|
|  86927|       3|
|  53963|       5|
|  69042|       4|
+-------+--------+
only showing top 20 rows
```

Command took 9.93 seconds -- by meghana.rwgsql@gmail.com at 8/30/2017, 9:32:12 PM on movies_project

Cmd 26

```
# Colaborative filtering
# Splitting Data into Train and test
(training, test) = ratingsDF.randomSplit([0.8, 0.2])
```

```
Command took 0.04 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 10:39:04 AM on movies_project
```

Cmd 27

```
training.show()
```

▸ (1) Spark Jobs

```
+------+-------+------+--------------------+
|userId|movieId|rating|           timestamp|
+------+-------+------+--------------------+
|     1|      2|   3.5|2005-04-02 23:53:...|
|     1|     29|   3.5|2005-04-02 23:31:...|
|     1|     47|   3.5|2005-04-02 23:32:...|
|     1|     50|   3.5|2005-04-02 23:29:...|
|     1|    112|   3.5|2004-09-10 03:09:...|
|     1|    151|   4.0|2004-09-10 03:08:...|
|     1|    253|   4.0|2005-04-02 23:35:...|
|     1|    260|   4.0|2005-04-02 23:33:...|
|     1|    293|   4.0|2005-04-02 23:31:...|
|     1|    296|   4.0|2005-04-02 23:32:...|
|     1|    318|   4.0|2005-04-02 23:33:...|
|     1|    367|   3.5|2005-04-02 23:53:...|
|     1|    541|   4.0|2005-04-02 23:30:...|
|     1|    593|   3.5|2005-04-02 23:31:...|
|     1|    653|   3.0|2004-09-10 03:08:...|
|     1|    919|   3.5|2004-09-10 03:07:...|
|     1|    924|   3.5|2004-09-10 03:06:...|
|     1|   1009|   3.5|2005-04-02 23:53:...|
|     1|   1036|   4.0|2005-04-02 23:44:...|
|     1|   1079|   4.0|2004-09-10 03:07:...|
+------+-------+------+--------------------+
only showing top 20 rows
```

```
Command took 15.41 seconds -- by meghana.rwgsql@gmail.com at 7/4/2017, 5:27:31 PM on Movies_Project
```

Cmd 28

```
#comparing Training and Test Dataset
tempDF = training.join(test, training.movieId == test.movieId)
display(tempDF)
```

▸ (1) Spark Jobs

userId	movieId	rating	timestamp
603	148	2	1996-07-25T15:28:36.000+0000
603	148	2	1996-07-25T15:28:36.000+0000
603	148	2	1996-07-25T15:28:36.000+0000
603	148	2	1996-07-25T15:28:36.000+0000
603	148	2	1996-07-25T15:28:36.000+0000
603	148	2	1996-07-25T15:28:36.000+0000
603	148	2	1996-07-25T15:28:36.000+0000

Showing the first 1000 rows.

```
Command took 50.24 seconds -- by meghana.rwgsql@gmail.com at 6/29/2017, 7:36:38 PM on
Movies_project
```

Cmd 29

```
#Dataset in Training and not in Test
filtered_test = training.subtract(test)

display (filtered_test)
filtered_test.count()
```

▸ (2) Spark Jobs

userId	movieId	rating
13	509	4
14	31658	4
21	953	3
21	1282	3
21	4034	4
24	2539	2
24	4226	4
26	587	4

Showing the first 1000 rows.

```
Command took 1.90 minutes -- by meghana.rwgsql@gmail.com at 6/29/2017, 8:08:32 PM on
Movies_project
```

Cmd 30

```
print "Training set size: ", training.count()
print "Test set size: ", test.count()
#print "Validation set size: ", test.count()
```

▸ (2) Spark Jobs

```
Training set size:  15998929
Test set size:  4001334
```

Command took 1.13 minutes -- by meghana.rwgsql@gmail.com at 8/31/2017, 10:39:13 AM on movies_project

Cmd 31

```
# Build the recommendation model using ALS on the training data
# Note we set cold start strategy to 'drop' to ensure we don't get NaN evaluation metrics
from pyspark.sql import SparkSession
from pyspark.ml.evaluation import RegressionEvaluator
from pyspark.ml.recommendation import ALS
from pyspark.sql import Row

als = ALS(rank=10, maxIter=10, regParam=0.01, userCol="userId", itemCol="movieId", ratingCol="rating")
#als = ALS(maxIter=5, regParam=0.01, userCol="userId", itemCol="movieId", ratingCol="rating", coldStartStrategy="drop")
model = als.fit(training)
```

▸ (5) Spark Jobs

Command took 2.11 minutes -- by meghana.rwgsql@gmail.com at 8/31/2017, 10:40:44 AM on movies_project

Cmd 32

```
#predictions = model.transform(test).dropna()
predictions = model.transform(test)
predictions = predictions.dropna()
predictions.registerTempTable("predictions_table")
```

Command took 0.12 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 10:42:57 AM on movies_project

Cmd 33

```
#%sql select user_id, movie_id, rating, prediction from predictions
spark.sql("select userId, movieId, rating, prediction from predictions_table").show()

```

▸ (1) Spark Jobs

```
+------+-------+------+----------+
|userId|movieId|rating|prediction|
+------+-------+------+----------+
| 92852|    148|   3.0| 2.5335486|
| 81218|    148|   1.0| 2.7732363|
| 91782|    148|   3.0| 2.9888206|
| 13170|    148|   3.0| 0.0466154|
|  1259|    148|   5.0| 3.2165732|
| 44882|    148|   4.0| 2.3504906|
| 94994|    148|   4.0|  3.119975|
| 90757|    148|   3.0| 2.9625764|
|  3673|    148|   2.0| 2.6216304|
| 64843|    148|   3.5|  2.630504|
| 81300|    148|   1.0| 2.3215954|
|118205|    148|   3.5| 2.8588994|
|109121|    148|   4.0|  3.454861|
| 68360|    148|   2.0|  2.942938|
|132268|    148|   2.0| 1.5295749|
| 30699|    148|   3.0| 2.1022847|
| 28361|    148|   4.0|  4.177149|
| 36723|    148|   1.0| 1.9845809|
|  9084|    148|   2.0| 3.0726907|
| 66709|    148|   5.0|  3.728733|
+------+-------+------+----------+
only showing top 20 rows
```

Command took 35.43 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 10:43:05 AM on movies_project

Cmd 34

```
#Generate top 10 movie recommendations for each user
userRecs = model.recommendForAllUsers(10)
userRecs.show(10, truncate = False)
```

▸ (1) Spark Jobs

```
+------+--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
--------------------------------------+
|userId|recommendations


                                      |
+------+--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
--------------------------------------+
|148   |[[128366,10.951867], [1830,10.697975], [40969,9.900242], [27108,9.73136
5], [76022,9.663902], [69310,9.3615885], [94074,9.253457], [80189,8.878327], [73
170,8.857643], [87719,8.633941]]        |
|463   |[[116155,7.8540316], [87884,7.6012306], [76022,7.536461], [69931,7.45520
8], [107780,7.1698666], [69666,7.0395384], [72045,6.9239545], [87040,6.9138904],
[117308,6.8977456], [109633,6.892449]]|
|471   |[[74159,9.095757], [116951,8.812828], [76022,8.2325325], [27108,7.477155
7], [69310,7.318554], [88092,7.197364], [73501,6.787854], [61913,6.649144], [921
69,6.524055], [39244,6.459038]]        |
|496   |[[73529,7.6900644], [87884,7.430202], [69464,7.1698527], [88092,7.154989
2], [72388,7.055085], [116951,6.789952], [86572,6.6127315], [8411,6.582972], [84
796,6.551797], [73139,6.5232415]]      |
|833   |[[116951,13.432829], [61913,11.178102], [98126,10.112643], [74159,9.9142
13], [27108,9.62832], [104861,9.517236], [81075,9.331002], [26027,9.306303], [33
132,9.176152], [81059,9.121372]]       |
|1088  |[[86947,10.466293], [72388,10.142122], [92696,8.911163], [110380,8.63503
4], [104390,8.579515], [97715,8.305207], [44861,8.267274], [629,8.1867485], [777
19,8.138725], [55856,8.128115]]        |
|1238  |[[34729,7.121804], [27108,6.643126], [69931,6.5108547], [88092,6.18079
3], [73879,6.144084], [73529,6.084306], [67894,6.067928], [86947,6.0549393], [11
0380,6.03752], [82304,6.0205054]]        |
|1342  |[[6674,8.4584055], [27679,8.04373], [68976,7.850332], [72360,7.8449745],
[98595,7.82905], [72735,7.717164], [92169,7.561826], [8954,7.550848], [99387,7.5
01115], [57478,7.414889]]             |
|1580  |[[1830,16.55101], [128366,13.99382], [73170,13.127473], [40969,11.811542
5], [53548,11.645727], [116155,11.541295], [80189,11.204805], [92314,10.934051
5], [3209,10.876148], [82061,10.749665]] |
|1591  |[[53476,9.660441], [39244,9.43916], [87040,9.315279], [74159,9.262892],
 [66579,9.137938], [59302,8.514868], [76022,8.449947], [82261,8.342544], [82842,
8.307518], [77433,8.128576]]           |
+------+------------------------------------------------------------------------
--------------------------------------------------------------------------------
----------------------------------------+
only showing top 10 rows
```

Command took 56.02 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 11:03:41 AM on movies_project

Cmd 35

```
# Generate top 10 user recommendations for each movie
movieRecs = model.recommendForAllItems(10)
movieRecs.show(10,truncate = False)
```

▸ (1) Spark Jobs

```
+--------+----------------------------------------------------------------------
--------------------------------------------------------------------------------
-------------------------------------+
|movieId|recommendations


                                      |
+--------+----------------------------------------------------------------------
--------------------------------------------------------------------------------
```

```
------------------------------------+
|148    |[[33314,8.701766], [1425,8.180933], [81312,7.373311], [68156,7.208327
3], [106990,6.8708425], [86490,6.8576126], [131667,6.6769357], [113811,6.57130
2], [42589,6.48554], [36374,6.4662094]]  |
|463    |[[120759,7.827287], [103795,7.3661604], [1425,7.3247886], [7637,7.02861
55], [41182,6.988009], [65884,6.946594], [46236,6.7009373], [101608,6.600505],
 [117304,6.5729437], [130899,6.54625]] |
|471    |[[15629,7.3206534], [39244,7.006679], [68156,6.9869514], [55809,6.80674
6], [35429,6.7478523], [17999,6.739544], [107667,6.633784], [132964,6.6239376],
 [112794,6.5889277], [54192,6.460428]]|
|496    |[[1425,11.919976], [33314,8.572311], [42589,8.134817], [120759,7.643493
7], [101255,7.342348], [26714,7.2738714], [86490,7.260195], [59542,7.174942], [7
2203,7.1624765], [36374,7.149855]]   |
|833    |[[7245,6.9702134], [138324,6.848807], [86490,6.721907], [121230,6.65824
5], [126728,6.6005583], [29030,6.476948], [29388,6.4507437], [114637,6.3349886],
[51105,6.30784], [67679,6.2839894]] |
|1088   |[[61007,9.486544], [92390,8.747037], [7332,7.796366], [101608,7.43559
2], [95345,7.182036], [99906,7.117282], [8892,6.961412], [65884,6.9582014], [132
685,6.84749], [116845,6.832096]]       |
|1238   |[[61315,7.817972], [81312,6.8869853], [101255,6.847694], [110290,6.7657
6], [14403,6.6632776], [105116,6.627554], [27841,6.5338273], [7326,6.5334315],
 [92724,6.456632], [73969,6.453871]]   |
|1342   |[[1425,7.136017], [125248,6.5628214], [97609,6.553137], [122900,6.55167
8], [49826,6.5504475], [11941,6.4981246], [22946,6.486262], [124859,6.455388],
 [9635,6.372327], [9340,6.367054]]     |
|1580   |[[36089,6.003181], [861,5.9432993], [112095,5.7447734], [46278,5.713271
6], [127916,5.7111936], [71611,5.683817], [117250,5.68226], [115837,5.658524],
 [52622,5.6481333], [22718,5.6437683]] |
|1591   |[[116848,6.179776], [1425,6.070424], [35823,6.035642], [69716,5.94612
4], [96611,5.924816], [95981,5.9094105], [86490,5.883312], [112156,5.8413234],
 [28134,5.7672577], [27735,5.762802]]    |
+-------+-----------------------------------------------------------------------
--------------------------------------------------------------------------------
------------------------------------+
only showing top 10 rows
```

Command took 51.01 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 11:02:35 AM on movies_project

Cmd 36

```
spark.sql("select count(userId) as Users, movieId from predictions_table
group by movieId LIMIT 20").show()
```

▸ (1) Spark Jobs

```
+-----+-------+
|Users|movieId|
+-----+-------+
|   67|    148|
|   76|    463|
| 2253|    471|
|   80|    496|
|  283|    833|
| 2161|   1088|
|  598|   1238|
|  653|   1342|
| 7026|   1580|
| 1128|   1591|
| 2308|   1645|
|   52|   1829|
| 1023|   1959|
|  498|   2122|
|  424|   2142|
| 1291|   2366|
|   42|   2659|
|  258|   2866|
| 2697|   3175|
|   22|   3749|
+-----+-------+


Command took 38.77 seconds -- by meghana.rwgsql@gmail.com at 8/30/2017, 9:41:37 PM on
movies_project
```

Cmd 37

```python
from pyspark.sql import functions as F

# Rename column from movieId to ID
movies_DF = moviesDF.withColumnRenamed('movieId', 'ID')

# From predictions, create a movie_ids_with_avg_ratings_df that combines the two DataFrames
movie_ids_with_avg_predicted_ratings_df = predictions.groupBy('movieId').agg(F.count(predictions.prediction).alias("count"), F.avg(predictions.prediction).alias("average"))
#print 'movie_ids_with_avg_predicted_ratings_df:'
#movie_ids_with_avg_predicted_ratings_df.show(3, truncate=False)

# Note: movie_names_df is a temporary variable, used only to separate the steps necessary
# to create the movie_names_with_avg_ratings_df DataFrame.
#movie_names_df = movie_ids_with_avg_ratings_df.<FILL_IN>  # use df.join(df2, df.name == df2.name)
predicted_movie_names_df = movie_ids_with_avg_predicted_ratings_df.join(movies_DF, movie_ids_with_avg_predicted_ratings_df.movieId == movies_DF.ID)

#movie_names_with_avg_ratings_df = movie_names_df.<FILL_IN>
movie_names_with_avg_pred_ratings_df = predicted_movie_names_df.select("average", "title", "count", "movieId")

#print 'movie_names_with_avg_ratings_df:'
#movie_names_with_avg_pred_ratings_df.show(3, truncate=False)

#We want to filter our movies with high ratings but greater than or equal to 500 reviews.
#movies_with_500_ratings_or_more = movie_names_with_avg_ratings_df.<FILL_IN>
predicted_movies_with_500_ratings_or_more = (movie_names_with_avg_pred_ratings_df
                                   .where("count >= 500")
                                   
.orderBy(movie_names_with_avg_pred_ratings_df.average.desc()))

print 'Movies with highest ratings:'
predicted_movies_with_500_ratings_or_more.show(20, truncate=False)
```

▸ (2) Spark Jobs

```
movie_ids_with_avg_predicted_ratings_df:
Movies with highest ratings:
+------------------+----------------------------------------------------------
```

```
---------------+-----+-------+
|average           |title
               |count|movieId|
+------------------+------------------------------------------------------------
---------------+-----+-------+
|4.276359616133422 |Shawshank Redemption, The (1994)
               |12513|318    |
|4.240856685485456 |Third Man, The (1949)
               |1317 |1212   |
|4.219244047411683 |Godfather, The (1972)
               |8345 |858    |
|4.193616895280234 |Usual Suspects, The (1995)
               |9424 |50     |
|4.185131523330215 |Seven Samurai (Shichinin no samurai) (1954)
               |2359 |2019   |
|4.183755558500053 |Rear Window (1954)
               |3538 |904    |
|4.181936215899116 |Casablanca (1942)
               |4949 |912    |
|4.180760424454526 |Sunset Blvd. (a.k.a. Sunset Boulevard) (1950)
               |1322 |922    |
|4.164261522881997 |Dr. Strangelove or: How I Learned to Stop Worrying and Love
 the Bomb (1964)|4591 |750    |
|4.159615292880716 |Schindler's List (1993)
               |9804 |527    |
|4.157970987149139 |Paths of Glory (1957)
               |737  |1178   |
|4.151550216978014 |North by Northwest (1959)
               |3161 |908    |
|4.147988171927924 |Notorious (1946)
               |966  |930    |
|4.147914183192143 |Godfather: Part II, The (1974)
               |5466 |1221   |
|4.147455377140265 |Big Sleep, The (1946)
               |1131 |1284   |
|4.145923303171358 |Double Indemnity (1944)
               |963  |3435   |
|4.145621530264448 |12 Angry Men (1957)
               |2649 |1203   |
|4.141916454161409 |Rashomon (Rashômon) (1950)
               |738  |5291   |
|4.137867432680226 |Band of Brothers (2001)
               |899  |7502   |
|4.1342800594020535|Touch of Evil (1958)
               |925  |1248   |
+------------------+------------------------------------------------------------
---------------+-----+-------+
only showing top 20 rows
```

```
Command took 39.79 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 11:13:38 AM on
movies_project
```

Cmd 38

```
spark.sql("select count(*)from predictions_table").show()
```

▸ (1) Spark Jobs

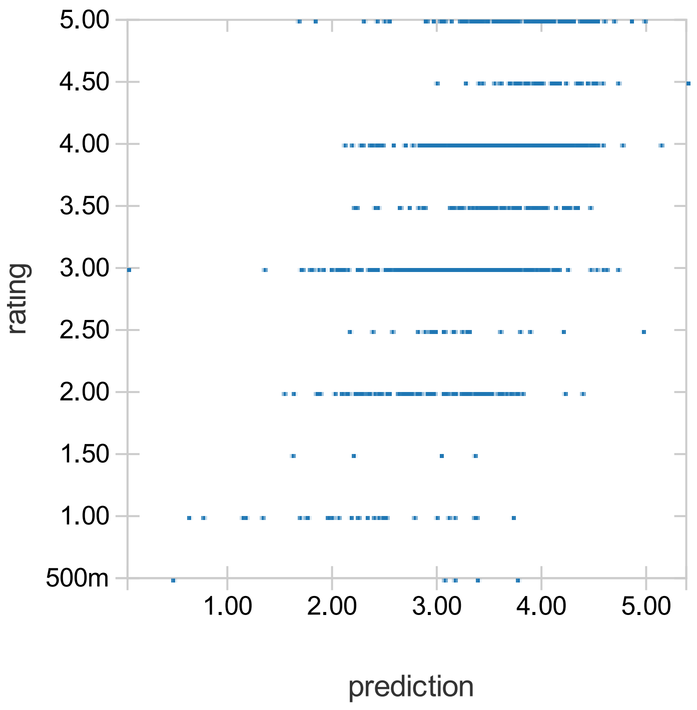
```
+---------+
|count(1)|
+---------+
| 4001283|
+---------+
```

```
Command took 39.19 seconds -- by meghana.rwgsql@gmail.com at 8/30/2017, 4:23:28 PM on
Movies_Recommendation
```

Cmd 39

```
display (spark.sql("select rating, prediction from predictions_table"))
```

▸ (1) Spark Jobs



Showing sample based on the first 1000 rows.

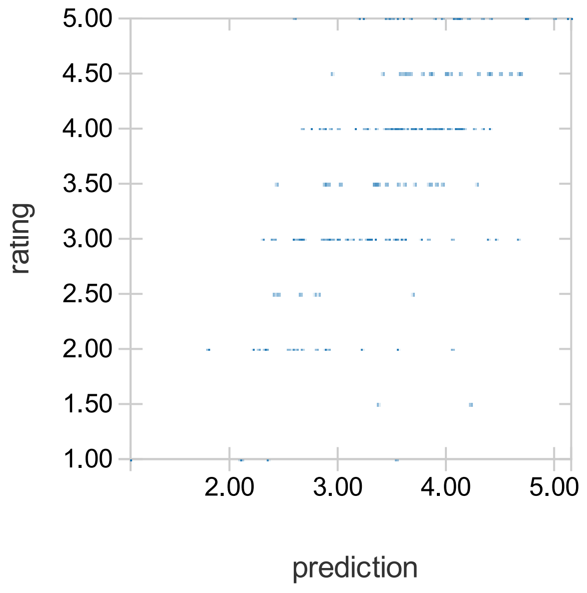
```
Command took 35.01 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 11:16:26 AM on
movies_project
```

Cmd 40

```

display(predictions.sample(False, 1000.0/ratingsDF.count()))
```

▸ (4) Spark Jobs



```
Command took 46.74 seconds -- by meghana.rwgsql@gmail.com at 8/30/2017, 9:59:47 PM on movies_project
```

Cmd 41

```
 # Evaluate the model by computing the RMSE on the test data
evaluator = RegressionEvaluator(metricName="rmse", labelCol="rating",
                                predictionCol="prediction")
rmse = evaluator.evaluate(predictions)
print("Root-mean-square error = " + str(rmse))
```

▸ (1) Spark Jobs

```
Root-mean-square error = 0.804667235236
```

```
Command took 39.97 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 11:17:09 AM on movies_project
```

Cmd 42

```
displayHTML("<h4>The Root-mean-square error is %s</h4>" % str(rmse))
```

The Root-mean-square error is 0.804667235236

```
Command took 0.04 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 11:17:59 AM on movies_project
```

Cmd 43

```
Movies_recommended = spark.sql("SELECT predictions_table.movieId, movies.title, rating, prediction \
                             FROM predictions_table JOIN movies on movies.movieId = predictions_table.movieId  \
                             Group by predictions_table.movieId, movies.title, rating, prediction \
                             Order by prediction DESC, rating DESC LIMIT 20")
Movies_recommended.show(truncate = False)
```

▸ (2) Spark Jobs

```
+--------+---------------------------------------+------+----------+
|movieId|title                                  |rating|prediction|
+--------+---------------------------------------+------+----------+
|3455   |Buddy Boy (1999)                       |1.0   |9.837434  |
|97059  |Katy Perry: Part of Me (2012)          |0.5   |9.402147  |
|99861  |Jesse Stone: Sea Change (2007)         |5.0   |9.29036   |
|26136  |Hallelujah Trail, The (1965)           |2.5   |9.207833  |
|119145 |Kingsman: The Secret Service (2015)    |1.0   |8.797971  |
|1165   |Bloody Child, The (1996)               |5.0   |8.539613  |
|46848  |Gumball Rally, The (1976)              |4.5   |8.5312395 |
|39416  |Kids in America (2005)                 |5.0   |8.268056  |
|2710   |Blair Witch Project, The (1999)        |3.0   |8.222418  |
|63992  |Twilight (2008)                        |5.0   |8.204944  |
|78772  |Twilight Saga: Eclipse, The (2010)     |5.0   |8.2048855 |
|71141  |Airbag (1997)                          |0.5   |8.130741  |
|501    |Naked (1993)                           |5.0   |8.091531  |
|1354   |Breaking the Waves (1996)              |5.0   |8.080061  |
|4491   |Criminal Law (1988)                    |0.5   |8.074579  |
|80505  |2012: Supernova (2009)                 |4.0   |8.061749  |
|7318   |Passion of the Christ, The (2004)      |5.0   |8.057657  |
|5922   |Attila (Attila Flagello di Dio) (1982)|4.5   |8.050166  |
|67501  |Kogel mogel (1988)                     |3.0   |8.045789  |
|26339  |Dolemite (1975)                        |5.0   |8.014356  |
+--------+---------------------------------------+------+----------+
```

```
Command took 44.85 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 11:18:10 AM on movies_project
```

Cmd 44

```
#Recommended movies
Movies_Recommended_from_model = spark.sql("SELECT
predictions_table.movieId, movies.title, avg(prediction) as
Predicted_ratings \
                                          FROM predictions_table \
                                          JOIN movies ON
movies.movieId=predictions_table.movieId \
                                          GROUP BY
predictions_table.movieId, movies.title  \
                                          ORDER BY AVG(prediction) DESC\
                                          LIMIT 20")
Movies_Recommended_from_model.show(truncate = False)
```

▸ (2) Spark Jobs

```
+--------+--------------------------------------------------------------------------------
--------------------------------+------------------+
|movieId|title
                                |Predicted_ratings |
+--------+--------------------------------------------------------------------------------
--------------------------------+------------------+
|107357 |Call Me Crazy: A Five Film (2013)
                                |7.227933883666992 |
|56022  |Harrison Bergeron (1995)
                                |6.917505264282227 |
|55659  |Return to the 36th Chamber (Shao Lin da peng da shi) (1980)
                                |6.843296051025391 |
|97196  |Shottas (2002)
                                |6.524924039840698 |
|80324  |Abandoned (2010)
                                |6.486706256866455 |
|89961  |Play (2011)
                                |6.443424701690674 |
|95021  |Outer Space (2000)
                                |6.418587684631348 |
|98834  |Fitzgerald Family Christmas, The (2012)
                                |6.3923845291137695|
|98755  |1911 (Xinhai geming) (2011)
                                |6.2648186683654785|
|78332  |Beautiful Person, The (La belle personne) (2008)
                                |6.190258026123047 |
|109416 |Bring It On: Fight to the Finish (2009)
                                |6.165205478668213 |
|96473  |Prime Suspect: Inner Circles (1995)
                                |6.139786720275879 |
|41650  |Mother India (1957)
                                |6.084947109222412 |
|49872  |Loose Change: Second Edition (2006)
                                |5.997299671173096 |
|74937  |Two-Minute Warning (1976)
                                |5.947014331817627 |
|84506  |Silent Souls (Ovsyanki) (2010)
                                |5.800346374511719 |
|99861  |Jesse Stone: Sea Change (2007)
                                |5.757004082202911 |
|27372  |Uprising (2001)
                                |5.701282978057861 |
|7441   |Thousand Clouds of Peace, A (Mil nubes de paz cercan el cielo, amor, ja
más acabarás de ser amor) (2003)|5.665889501571655 |
|81906  |Snow and Ashes (2010)
                                |5.659394264221191 |
+-------+-----------------------------------------------------------------------
--------------------------------+------------------+
```

Command took 41.43 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 11:19:31 AM on movies_project

Cmd 45

```
predictions.describe().show()
```

▸ (1) Spark Jobs

```
+-------+-----------------+------------------+------------------+---------------
---+
|summary|           userId|           movieId|            rating|        predict
ion|
+-------+-----------------+------------------+------------------+---------------
---+
|  count|          4000316|           4000316|           4000316|           4000
316|
|   mean|69017.89363490284|  9013.73503618214| 3.525570604922211|3.4412456563136
664|
| stddev|40043.73561717234|19720.687189242326|1.0519828325185538|0.7479698634538
245|
|    min|                1|                 1|               0.5|         -6.137
381|
|    max|           138493|            131013|               5.0|          9.837
434|
+-------+-----------------+------------------+------------------+---------------
---+
```

Command took 38.73 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 11:20:32 AM on movies_project

Cmd 46

```
moviesRDD=moviesDF.rdd
ratingsRDD=ratingsDF.rdd
```

Command took 0.12 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 12:47:08 PM on movies_project

Cmd 47

```
type(moviesRDD)
type(ratingsRDD)
```

```
Out[50]: pyspark.rdd.RDD
```

Command took 0.07 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 12:47:11 PM on movies_project

Cmd 48

```
from pyspark import SparkConf, SparkContext

numRatings  = ratingsRDD.count()
numUsers    = ratingsRDD.map(lambda r:r[0]).distinct().count()
numMovies   = ratingsRDD.map(lambda r:r[1]).distinct().count()
print "--- %d ratings from %d users for %d movies\n" % (numRatings,
numUsers, numMovies)
```

▸ (3) Spark Jobs

```
--- 20000263 ratings from 138493 users for 26744 movies
```

Command took 9.94 minutes -- by meghana.rwgsql@gmail.com at 8/30/2017, 5:34:53 PM on Movies_Recommendation

Cmd 49

```
#Adding new user ratings.
#Now we need to rate some movies for the new user. We will put them in a
new RDD and we will use the user ID 0, that is not assigned in the
MovieLens dataset

new_user_ID = 0
# The format of each line is (userID, movieID, rating)
new_user_ratings = [
(0,318,4.5,1112484580), # Shawshank Redemption (1994)
(0,858,4.7,1112484940), # Godfather (1972)
(0,50,5.0,1094785709), # Usual suspects (1995)
(0,527,5.0,1094785691), # Schindlers List (1993)
(0,1221,5.0,1094785759), # Godfather: Part II (1974)
(0,2019,4.8,1112484735), # Seven Samurai (1954)
(0,904,4.5,1094786062), # Rear window (1954)
(0,7502,4.7,1094785764), # Band of Brothers (2001)
(0,912,4.2,1112486150) , # Casablanca (1942)
(0,922,4.0,1112486098) # Sunset blvd (1950)
]
new_user_ratings_RDD = sc.parallelize(new_user_ratings)
print 'New user ratings: %s' % new_user_ratings_RDD.take(10)

```

▸ (3) Spark Jobs

```
New user ratings: [(0, 318, 4.5, 1112484580), (0, 858, 4.7, 1112484940), (0, 50,
5.0, 1094785709), (0, 527, 5.0, 1094785691), (0, 1221, 5.0, 1094785759), (0, 201
9, 4.8, 1112484735), (0, 904, 4.5, 1094786062), (0, 7502, 4.7, 1094785764), (0,
 912, 4.2, 1112486150), (0, 922, 4.0, 1112486098)]
```

Command took 0.47 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 12:47:23 PM on movies_project

Cmd 50

```
new_user_ratings =
new_user_ratings_RDD.toDF(['userId','movieId','rating','timestamp'])
```

▸ (1) Spark Jobs

Command took 0.12 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 12:49:31 PM on movies_project

Cmd 51

```
new_user_ratings = new_user_ratings.withColumn('rating', new_user_ratings["rating"].cast("float"))
new_user_ratings = new_user_ratings.withColumn('userId', new_user_ratings["userId"].cast("int"))
new_user_ratings = new_user_ratings.withColumn('movieId', new_user_ratings["movieId"].cast("int"))
new_user_ratings.show()
```

▸ (3) Spark Jobs

```
+------+-------+------+----------+
|userId|movieId|rating| timestamp|
+------+-------+------+----------+
|     0|    318|   4.5|1112484580|
|     0|    858|   4.7|1112484940|
|     0|     50|   5.0|1094785709|
|     0|    527|   5.0|1094785691|
|     0|   1221|   5.0|1094785759|
|     0|   2019|   4.8|1112484735|
|     0|    904|   4.5|1094786062|
|     0|   7502|   4.7|1094785764|
|     0|    912|   4.2|1112486150|
|     0|    922|   4.0|1112486098|
+------+-------+------+----------+
```

Command took 0.22 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 12:49:35 PM on movies_project

Cmd 52

```
from pyspark.sql.functions import from_unixtime
new_user_ratings = new_user_ratings.withColumn('timestamp', from_unixtime(new_user_ratings.timestamp))
```

Command took 0.04 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 12:49:44 PM on movies_project

Cmd 53

```
from pyspark.sql.types import TimestampType
new_user_ratings = new_user_ratings.withColumn('timestamp', new_user_ratings.timestamp.cast(TimestampType()))
```

Command took 0.07 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 12:49:47 PM on movies_project

Cmd 54

```
new_user_ratings.printSchema()
```

```
root
 |-- userId: integer (nullable = true)
 |-- movieId: integer (nullable = true)
 |-- rating: float (nullable = true)
 |-- timestamp: timestamp (nullable = true)
```

Command took 0.04 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 12:49:51 PM on movies_project

Cmd 55

```
training_with_my_ratings = training.unionAll(new_user_ratings)
```

Command took 0.07 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 12:50:58 PM on movies_project

Cmd 56

```
display(new_user_ratings.limit(10))
```

▸ (3) Spark Jobs

userId	movieId	rating
0	318	4.5
0	858	4.7
0	50	5
0	527	5
0	1221	5
0	2019	4.8
0	904	4.5
0	7502	4.7
0	912	4.2
0	922	4

Command took 0.22 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 1:24:53 PM on movies_project

Cmd 57

```
# TODO: Replace <FILL IN> with appropriate code
training_with_my_ratings = training.unionAll(new_user_ratings)

print ('The training dataset now has %s more entries than the original training dataset' %
       (training_with_my_ratings.count() - training.count()))
assert (training_with_my_ratings.count() - training.count()) == new_user_ratings.count()
#The training dataset now has 11 more entries than the original training dataset
```

▸ (5) Spark Jobs

```
The training dataset now has 10 more entries than the original training dataset
```

Command took 1.96 minutes -- by meghana.rwgsql@gmail.com at 8/31/2017, 1:21:36 PM on movies_project

Cmd 58

```
#Training  new dataset with ALS model

# Build the new recommendation model using ALS on the training data
# Note we set cold start strategy to 'drop' to ensure we don't get NaN evaluation metrics
from pyspark.sql import SparkSession
from pyspark.ml.evaluation import RegressionEvaluator
from pyspark.ml.recommendation import ALS
#from pyspark.mllib.recommendation import ALS, Rating
from pyspark.sql import Row

als = ALS(rank=10, maxIter=10, regParam=0.01, userCol="userId", itemCol="movieId", ratingCol="rating")
#als = ALS(maxIter=5, regParam=0.01, userCol="userId", itemCol="movieId", ratingCol="rating", coldStartStrategy="drop")
new_rating_model = als.fit(training_with_my_ratings)

#print "New model trained in %s seconds" % round(tt,3)

```

▸ (5) Spark Jobs

Command took 2.10 minutes -- by meghana.rwgsql@gmail.com at 8/31/2017, 12:51:16 PM on movies_project

Cmd 59

```
#predictions = model.transform(test).dropna()
predictions_with_my_ratings = new_rating_model.transform(test)
predictions_with_my_ratings = predictions_with_my_ratings.dropna()
predictions_with_my_ratings.registerTempTable("predictions_with_my_ratings_table")
```

Command took 0.07 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 1:28:27 PM on movies_project

Cmd 60

```
spark.sql("select * from predictions_with_my_ratings_table order by prediction desc LIMIT 10").show()

```

▸ (1) Spark Jobs

```
+------+-------+------+-------------------+----------+
|userId|movieId|rating|          timestamp|prediction|
+------+-------+------+-------------------+----------+
| 24994| 100010|   4.0|2015-02-20 22:19:47| 10.508722|
|138215|    876|   5.0|2002-11-10 06:20:00| 10.221236|
| 51007|   3228|   2.0|2000-08-28 20:06:50|  9.273807|
| 47938|  67501|   3.0|2010-02-06 19:16:39|  9.200949|
|119833| 111320|   5.0|2014-07-31 03:24:01|  8.994706|
|117621|   4244|   5.0|2007-03-28 09:35:20|  8.776396|
| 78022|   7441|   3.5|2004-08-10 03:14:23|  8.760872|
| 24480|   5922|   4.5|2006-07-12 03:20:59|  8.693072|
|104117|  56093|   5.0|2010-02-07 09:21:14|  8.650107|
|113325|  44595|   0.5|2010-12-17 07:36:11|  8.552686|
+------+-------+------+-------------------+----------+
```

```
Command took 37.57 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 2:54:54 PM on
movies_project
```

Cmd 61

```
# Generate top 10 user recommendations for each movie
movieRecsnew = new_rating_model.recommendForAllItems(10)
movieRecsnew.show(10,truncate = False)
```

▸ (1) Spark Jobs

```
+-------+---------------------------------------------------------------------------
-------------------------------------------------------------------------------------
----------------------------------------+
|movieId|recommendations


                                        |
+-------+---------------------------------------------------------------------------
-------------------------------------------------------------------------------------
----------------------------------------+
|148    |[[124002,8.557087], [89066,8.09482], [43815,7.5858655], [33314,7.49618
5], [128821,7.2536316], [56145,6.8170147], [112318,6.7748556], [5747,6.746767],
 [61007,6.746116], [87447,6.7400374]]      |
|463    |[[101608,7.5712504], [65884,7.5201693], [92390,7.3081822], [118248,7.15
46474], [79298,6.9978046], [7637,6.960828], [28022,6.9503565], [54584,6.917762
8], [67339,6.9005127], [112156,6.639238]]  |
|471    |[[107804,6.9638305], [68156,6.7373343], [75545,6.385509], [72458,6.2377
696], [35429,6.223061], [55716,6.198926], [34369,6.1974516], [53118,6.1938286],
 [107667,6.1792274], [120483,6.1639795]]  |
|496    |[[1425,8.402818], [5625,8.163208], [58845,8.043345], [84580,7.989399],
 [119988,7.8339767], [17816,7.8166494], [130693,7.7461033], [4273,7.668422], [11
2469,7.6296096], [74030,7.615759]]        |
|833    |[[97184,7.3606925], [75306,6.9260845], [138215,6.9125247], [67679,6.848
727], [5474,6.839116], [86166,6.765509], [5747,6.7330513], [121534,6.7165036],
 [121230,6.4955673], [50529,6.48988]]      |
|1088   |[[74576,7.644407], [133404,7.607803], [76772,7.459447], [120759,7.44451
9], [24829,7.4256725], [93821,7.417887], [62342,7.396893], [93809,7.390879], [72
45,7.360171], [110831,7.3552527]]        |
|1238   |[[61315,7.6330214], [53192,6.8270493], [19366,6.5271854], [5747,6.52172
1], [84889,6.5086412], [14403,6.4422936], [14571,6.4190235], [51055,6.383786],
 [87447,6.3422785], [9241,6.3329163]]      |
|1342   |[[97609,7.4944735], [124859,6.8256826], [133374,6.8231425], [63539,6.64
2704], [63122,6.6319804], [122900,6.5695662], [114432,6.5227604], [1425,6.513621
3], [68007,6.2603736], [115976,6.259104]]|
|1580   |[[115837,6.137461], [42714,5.9726377], [54192,5.9595323], [20349,5.8790
793], [72725,5.8467574], [34369,5.836102], [88922,5.812204], [36089,5.791688],
 [44957,5.7126417], [22718,5.70847]]       |
|1591   |[[1425,7.4258857], [138215,6.4615006], [86490,6.347809], [99021,6.15392
3], [112156,6.141174], [97136,6.1349545], [69716,6.087041], [27735,6.070429], [1
07650,5.999553], [54192,5.922649]]       |
+-------+------------------------------------------------------------------------
--------------------------------------------------------------------------------
-----------------------------------------+
only showing top 10 rows
```

Command took 58.34 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 2:50:58 PM on movies_project

Cmd 62

```
 # Evaluate the model by computing the RMSE on the test data
evaluator = RegressionEvaluator(metricName="rmse", labelCol="rating",
                                predictionCol="prediction")
rmse = evaluator.evaluate(predictions_with_my_ratings)
print("Root-mean-square error for new ratings = " + str(rmse))*
```

(1) Spark Jobs

```
Root-mean-square error for new ratings = 0.805098946653
```

Command took 38.66 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 1:30:03 PM on movies_project

Cmd 63

```
#PREDICT YOUR RATINGS
# Create a list of my rated movie IDs
my_rated_movie_ids = [x[1] for x in new_user_ratings]
my_rated_movie_ids
```

```
Out[89]:
[Column<userId[1]>,
 Column<movieId[1]>,
 Column<rating[1]>,
 Column<timestamp[1]>]
```

```
Command took 0.07 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 1:44:12 PM on movies_project
```

Cmd 64

```
moviesDF = moviesDF.withColumn('movieId', moviesDF["movieId"].cast("int"))
```

```
Command took 0.07 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 1:52:22 PM on movies_project
```

Cmd 65

```
moviesDF.printSchema()
```

```
root
 |-- movieId: integer (nullable = true)
 |-- title: string (nullable = true)
 |-- genres: string (nullable = true)
```

```
Command took 0.07 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 1:52:37 PM on movies_project
```

Cmd 66

```
# Filter out the movies I already rated.
not_rated_df = moviesDF.filter(~
moviesDF["movieId"].isin(my_rated_movie_ids))
```

```
AnalysisException: u"Can't extract value from userId#5274: need struct type but
 got int;"
```

```
Command took 0.17 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 1:53:43 PM on movies_project
```

Cmd 67

```
# Rename the "ID" column to be "movieId", and add a column with my_user_id
as "userId".
my_unrated_movies_df = not_rated_df.selectExpr("ID as
movieId").withColumn('userId', F.lit(my_user_id))

# Use my_rating_model to predict ratings for the movies that I did not
manually rate.
raw_predicted_ratings_df = my_ratings_model.transform(my_unrated_movies_df)

predicted_ratings_df =
raw_predicted_ratings_df.filter(raw_predicted_ratings_df['prediction'] !=
float('nan'))
```

Cmd 68

```
predicted_with_counts_df = 
predicted_ratings_df.join(movie_names_with_avg_ratings_df,movie_names_with_
avg_ratings_df["movieId"]==predicted_ratings_df["movieId"])
predicted_highest_rated_movies_df = 
predicted_with_counts_df.filter(predicted_with_counts_df["count"]>75).sort(
"prediction",ascending=False)

print ('My 25 highest rated movies as predicted (for movies with more than
75 reviews):')
predicted_highest_rated_movies_df.show(25)
```

Cmd 69

```python
from pyspark.sql import functions as F

# Rename column from movieId to ID
movies_DF = moviesDF.withColumnRenamed('movieId', 'ID')

# From predictions, create a movie_ids_with_avg_ratings_df that combines the two DataFrames
new_movie_ids_with_avg_predicted_ratings_df = predictions_with_my_ratings.groupBy('movieId').agg(F.count(predictions_with_my_ratings.prediction).alias("count"), F.avg(predictions_with_my_ratings.prediction).alias("average"))
#print 'movie_ids_with_avg_predicted_ratings_df:'
#movie_ids_with_avg_predicted_ratings_df.show(3, truncate=False)

# Note: movie_names_df is a temporary variable, used only to separate the steps necessary
# to create the movie_names_with_avg_ratings_df DataFrame.
#movie_names_df = movie_ids_with_avg_ratings_df.<FILL_IN>  # use df.join(df2, df.name == df2.name)
new_predicted_movie_names_df = new_movie_ids_with_avg_predicted_ratings_df.join(movies_DF, new_movie_ids_with_avg_predicted_ratings_df.movieId == movies_DF.ID)

#movie_names_with_avg_ratings_df = movie_names_df.<FILL_IN>
new_movie_names_with_avg_pred_ratings_df = new_predicted_movie_names_df.select("average", "title", "count", "movieId")

#print 'movie_names_with_avg_ratings_df:'
#movie_names_with_avg_pred_ratings_df.show(3, truncate=False)

#We want to filter our movies with high ratings but greater than or equal to 500 reviews.
#movies_with_500_ratings_or_more = movie_names_with_avg_ratings_df.<FILL_IN>
new_predicted_movies_with_500_ratings_or_more = (new_movie_names_with_avg_pred_ratings_df
                                    .where("count >= 500")

.orderBy(new_movie_names_with_avg_pred_ratings_df.average.desc()))

print 'Movies with highest ratings:'
new_predicted_movies_with_500_ratings_or_more.show(20, truncate=False)
```

▸ (2) Spark Jobs

```
Movies with highest ratings:
+-----------------+-------------------------------------------------------------
```

```
--------------+-----+-------+
|average          |title
              |count|movieId|
+-----------------+------------------------------------------------------------
--------------+-----+-------+
|4.254283139357548|Shawshank Redemption, The (1994)
              |12513|318    |
|4.22503672482484 |Third Man, The (1949)
              |1317 |1212   |
|4.195024231735141|Godfather, The (1972)
              |8345 |858    |
|4.175523698708929|Seven Samurai (Shichinin no samurai) (1954)
              |2359 |2019   |
|4.17406071068002 |Rear Window (1954)
              |3538 |904    |
|4.173247438199274|Usual Suspects, The (1995)
              |9424 |50     |
|4.172001171513332|Sunset Blvd. (a.k.a. Sunset Boulevard) (1950)
              |1322 |922    |
|4.170191933998413|Casablanca (1942)
              |4949 |912    |
|4.152172995341341|Dr. Strangelove or: How I Learned to Stop Worrying and Love t
he Bomb (1964)|4591 |750    |
|4.140182386042633|Schindler's List (1993)
              |9804 |527    |
|4.138639154275057|North by Northwest (1959)
              |3161 |908    |
|4.138031953557074|Rashomon (Rashômon) (1950)
              |738  |5291   |
|4.135877628541436|Big Sleep, The (1946)
              |1131 |1284   |
|4.134845475108205|Double Indemnity (1944)
              |963  |3435   |
|4.131054048945816|Paths of Glory (1957)
              |737  |1178   |
|4.130510358326207|12 Angry Men (1957)
              |2649 |1203   |
|4.127649324721303|Thin Man, The (1934)
              |638  |950    |
|4.126916712084419|Notorious (1946)
              |966  |930    |
|4.125684980212188|All About Eve (1950)
              |943  |926    |
|4.125092605320183|Touch of Evil (1958)
              |925  |1248   |
+-----------------+------------------------------------------------------------
--------------+-----+-------+
only showing top 20 rows
```

Command took 46.45 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 2:58:27 PM on movies_project

Cmd 70

```
display(new_predicted_movies_with_500_ratings_or_more.limit(10))
```

▸ (2) Spark Jobs

average	title
4.254283139357548	Shawshank Redemption, The (1994)
4.22503672482484	Third Man, The (1949)
4.195024231735141	Godfather, The (1972)
4.175523698708929	Seven Samurai (Shichinin no samurai) (1954)
4.17406071068002	Rear Window (1954)
4.173247438199274	Usual Suspects, The (1995)
4.172001171513332	Sunset Blvd. (a.k.a. Sunset Boulevard) (1950)
4.170191933998413	Casablanca (1942)
4.152172995341341	Dr. Strangelove or: How I Learned to Stop Worrying

Command took 48.04 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 3:58:20 PM on movies_project

Cmd 71

```
predictions_with_my_ratings.registerTempTable("predictions_with_my_ratings_table")
```

Cmd 72

```
new_predicted_movies_with_500_ratings_or_more.registerTempTable("Recommended_movies")
```

Command took 0.07 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 3:03:27 PM on movies_project

Cmd 73

```
spark.sql("select * from Recommended_movies limit 10").show(truncate = False)
```

▸ (2) Spark Jobs

```
+-----------------+------------------------------------------------------------------------------+-----+-------+
|average          |title                                                                         |count|movieId|
+-----------------+------------------------------------------------------------
--------------+-----+-------+
|4.254283139357548|Shawshank Redemption, The (1994)
              |12513|318    |
|4.22503672482484 |Third Man, The (1949)
              |1317 |1212   |
|4.195024231735141|Godfather, The (1972)
              |8345 |858    |
|4.175523698708929|Seven Samurai (Shichinin no samurai) (1954)
              |2359 |2019   |
|4.17406071068002 |Rear Window (1954)
              |3538 |904    |
|4.173247438199274|Usual Suspects, The (1995)
              |9424 |50     |
|4.172001171513332|Sunset Blvd. (a.k.a. Sunset Boulevard) (1950)
              |1322 |922    |
|4.170191933998413|Casablanca (1942)
              |4949 |912    |
|4.152172995341341|Dr. Strangelove or: How I Learned to Stop Worrying and Love t
he Bomb (1964)|4591 |750    |
|4.140182386042633|Schindler's List (1993)
              |9804 |527    |
+-----------------+------------------------------------------------------------
--------------+-----+-------+
```

Command took 38.34 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 3:07:26 PM on movies_project

Cmd 74

```
Rec_movies_gt_500_reviews = spark.sql("select Recommended_movies.movieId, Recommended_movies.title, Recommended_movies.count, Recommended_movies.average,predictions_with_my_ratings_table.prediction \
           FROM Recommended_movies \
           JOIN predictions_with_my_ratings_table ON predictions_with_my_ratings_table.movieId=Recommended_movies.movieId \
           ORDER BY predictions_with_my_ratings_table.prediction DESC\
           LIMIT 20")

Rec_movies_gt_500_reviews.show(truncate = False)
```

▸ (3) Spark Jobs

```
+-------+----------------------------------------------------------+-----+-------
-----------+----------+
|movieId|title                                                     |count|average
           |prediction|
+-------+----------------------------------------------------------+-----+-------
-----------+----------+
|1035   |Sound of Music, The (1965)                                |2818 |3.71041
7884368267 |7.885456  |
|1354   |Breaking the Waves (1996)                                 |756  |3.73973
69243322856|7.354319  |
|913    |Maltese Falcon, The (1941)                                |2451 |4.09777
60245974135|7.315525  |
|327    |Tank Girl (1995)                                          |1441 |2.80837
23716174127|7.1648254 |
|1923   |There's Something About Mary (1998)                       |4882 |3.43619
35991432175|7.1481495 |
|1721   |Titanic (1997)                                            |6576 |3.19622
90323677034|7.135368  |
|39     |Clueless (1995)                                           |5173 |3.35956
27621147495|7.1215887 |
|39     |Clueless (1995)                                           |5173 |3.35956
27621147495|7.068882  |
|2700   |South Park: Bigger, Longer and Uncut (1999)               |3442 |3.52184
4895531676 |7.043368  |
|231    |Dumb & Dumber (Dumb and Dumber) (1994)                    |6354 |2.86068
06995677223|7.019651  |
|1088   |Dirty Dancing (1987)                                      |2220 |3.16447
0894111169 |6.9840746 |
|1      |Toy Story (1995)                                          |9991 |3.77902
5166437064 |6.9619803 |
|34     |Babe (1995)                                               |6462 |3.54798
3020373987 |6.9606657 |
|920    |Gone with the Wind (1939)                                 |2895 |3.71598
33018489454|6.943983  |
|6503   |Charlie's Angels: Full Throttle (2003)                    |862  |2.46360
7352559742 |6.9119782 |
|899    |Singin' in the Rain (1952)                                |2064 |4.01147
9699966144 |6.868468  |
|5618   |Spirited Away (Sen to Chihiro no kamikakushi) (2001)      |2665 |4.09501
6690501576 |6.836734  |
|7153   |Lord of the Rings: The Return of the King, The (2003)     |6349 |3.96917
1227554878 |6.835625  |
|1721   |Titanic (1997)                                            |6576 |3.19622
90323677034|6.8279424 |
|4993   |Lord of the Rings: The Fellowship of the Ring, The (2001)|7490 |3.98471
8856287098 |6.8215823 |
+-------+---------------------------------------------------------+-----+-------
-----------+----------+
```

```
Command took 1.54 minutes -- by meghana.rwgsql@gmail.com at 8/31/2017, 3:21:29 PM on
movies_project
```

Cmd 75

```
display(Rec_movies_gt_500_reviews)
```

▸ (3) Spark Jobs

movieId	title
1035	Sound of Music, The (1965)
1354	Breaking the Waves (1996)
913	Maltese Falcon, The (1941)
327	Tank Girl (1995)
1923	There's Something About Mary (1998)
1721	Titanic (1997)
39	Clueless (1995)
39	Clueless (1995)
2700	South Park: Bigger, Longer and Uncut (1999)

```
Command took 1.78 minutes -- by meghana.rwgsql@gmail.com at 8/31/2017, 3:53:03 PM on
movies_project
```

Cmd 76

```
#Saving the model
from pyspark.mllib.recommendation import MatrixFactorizationModel
import os

## Mount S3 bucket nycdsabootcamp to the Databricks File System
s3Path = "s3a://{0}:{1}@{2}".format("AKIAI2P5MSEO2JYXJVQQ",

"YJboxXSbraX4rg17aqtI+HmBjWCcpu4dxv2HW+bm",
                                    "nycdsabootcamp")
mntPath = "/mnt/data/"
model_path = os.path.join('..', 'models', 'movie_lens_als')
# Save and load model
model.save(model_path)
same_model = MatrixFactorizationModel.load(sc, model_path)

```

```
IllegalArgumentException: u'Path must be absolute: dbfs:/../models/movie_lens_al
s'
```

```
Command took 0.13 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 3:50:20 PM on
movies_project
```

Cmd 77

```
#MODEL TUNING AND CROSS VALIDATION

from pyspark.ml.tuning import CrossValidator, ParamGridBuilder

# 1. you first need to build a parameter grid based on your ALS model

paramGrid = ParamGridBuilder() \
                    .addGrid(als.rank, [10, 30]) \
                    .addGrid(als.maxIter, [10, 20]) \
                    .build()

# 2. use your ALS estimator and parameter grid to build cross validator
crossval = CrossValidator(estimator=als,
                          estimatorParamMaps=paramGrid,
                          evaluator=evaluator,
                          numFolds=3)  # use 3+ folds in practice

# Run cross-validation, and choose the best set of parameters.
cvModel = crossval.fit(training)
```

▸ (6) Spark Jobs

```
Command took 1.11 hours -- by meghana.rwgsql@gmail.com at 8/31/2017, 11:29:07 AM on
movies_project
```

Cmd 78

```
# Make predictions on test documents. cvModel uses the best model
cv_predictions = cvModel.transform(test)
```

```
Command took 0.07 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 12:38:19 PM on
movies_project
```

Cmd 79

```
cv_predictions.select("userId", "movieId", "rating", "prediction").show(10)

```

▸ (1) Spark Jobs

```
+------+-------+------+----------+
|userId|movieId|rating|prediction|
+------+-------+------+----------+
| 92852|    148|   3.0| 2.5335486|
| 81218|    148|   1.0| 2.7732363|
| 91782|    148|   3.0| 2.9888206|
| 13170|    148|   3.0| 0.0466154|
|  1259|    148|   5.0| 3.2165732|
| 44882|    148|   4.0| 2.3504906|
| 94994|    148|   4.0|  3.119975|
| 90757|    148|   3.0| 2.9625764|
|  3673|    148|   2.0| 2.6216304|
| 64843|    148|   3.5|  2.630504|
+------+-------+------+----------+
only showing top 10 rows
```

```
Command took 34.71 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 12:38:24 PM on
movies_project
```

Cmd 80

```
cv_predictions = cv_predictions.dropna()
cv_predictions.registerTempTable("cv_predictions_table")
```

```
Command took 0.07 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 12:39:12 PM on
movies_project
```

Cmd 81

```
#%sql select user_id, movie_id, rating, prediction from predictions
spark.sql("select userId, movieId, rating, prediction from
cv_predictions_table").show()
```

▸ (1) Spark Jobs

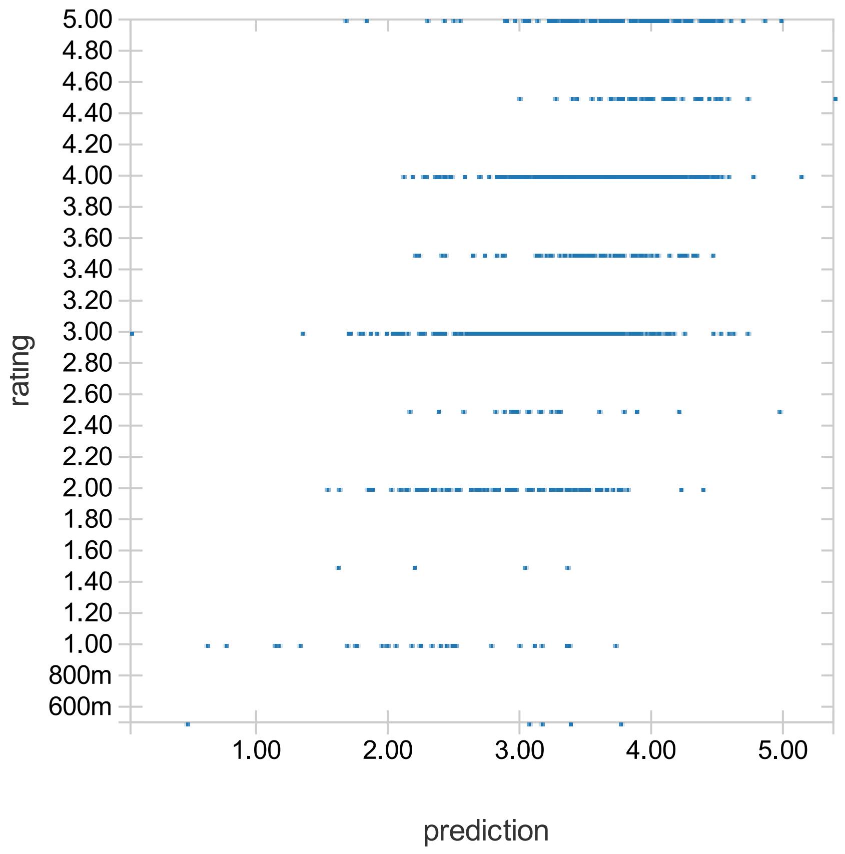
```
+------+-------+------+----------+
|userId|movieId|rating|prediction|
+------+-------+------+----------+
| 92852|    148|   3.0| 2.5335486|
| 81218|    148|   1.0| 2.7732363|
| 91782|    148|   3.0| 2.9888206|
| 13170|    148|   3.0| 0.0466154|
|  1259|    148|   5.0| 3.2165732|
| 44882|    148|   4.0| 2.3504906|
| 94994|    148|   4.0|  3.119975|
| 90757|    148|   3.0| 2.9625764|
|  3673|    148|   2.0| 2.6216304|
| 64843|    148|   3.5|  2.630504|
| 81300|    148|   1.0| 2.3215954|
|118205|    148|   3.5| 2.8588994|
|109121|    148|   4.0|  3.454861|
| 68360|    148|   2.0|  2.942938|
|132268|    148|   2.0| 1.5295749|
| 30699|    148|   3.0| 2.1022847|
| 28361|    148|   4.0|  4.177149|
| 36723|    148|   1.0| 1.9845809|
|  9084|    148|   2.0| 3.0726907|
| 66709|    148|   5.0|  3.728733|
+------+-------+------+----------+
only showing top 20 rows
```

```
Command took 35.14 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 12:40:37 PM on movies_project
```

Cmd 82

```
display (spark.sql("select rating, prediction from cv_predictions_table"))
```

▸ (1) Spark Jobs



Showing sample based on the first 1000 rows.

```
Command took 34.90 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 12:41:18 PM on movies_project
```

Cmd 83

```
#Calculating root mean square
cv_rmse = evaluator.evaluate(cv_predictions)
print("Root-mean-square error = " + str(cv_rmse))
```

▸ (1) Spark Jobs

```
Root-mean-square error = 0.804667235236
```

```
Command took 38.75 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 12:42:07 PM on movies_project
```

Cmd 84

```
displayHTML("<h4>The Root-mean-square error is %s</h4>" % str(cv_rmse))
```

The Root-mean-square error is 0.804667235236

```
Command took 0.07 seconds -- by meghana.rwgsql@gmail.com at 8/31/2017, 12:43:23 PM on
movies_project
```

```
Cmd 85
```

```
Cmd 86
```

▸ (2) Spark Jobs

```
+-------+--------------------+--------------------+
|movieId|               title|cv_Predicted_ratings|
+-------+--------------------+--------------------+
| 129401|Kevin Smith: Sold...|  5.5531861782073975|
|  79507|Amar Akbar Anthon...|   5.346615791320801|
| 111329| Memorial Day (2011)|  5.2839155197143555|
| 110173|         Wolf (2013)|   5.074623107910156|
|  69609|        Holly (2006)|   5.021418571472168|
|  78064|Ween Live in Chic...|   5.015072345733643|
| 116155|   Still Life (2013)|   4.957600116729736|
|  93729|      Pageant (2008)|   4.924448490142822|
|  26433|      Rockers (1978)|   4.919745445251465|
|  72866|Sandra of a Thous...|   4.883637428283691|
|  26968|  Cremaster 5 (1997)|   4.876115322113037|
|  73139|Out 1: Spectre (1...|   4.826381206512451|
| 112577|Willie & Phil (1980)|   4.805362701416016|
|  79842|     For Neda (2010)|   4.802628517150879|
|  98221|Year One, The (L'...|   4.788150787353516|
| 122290|      Homeboy (1988)|   4.785171031951904|
|  26109|Crooks in Clover ...|   4.757693290710449|
| 123107|The Phantom of th...|  4.7281880378723145|
|  84248|Organizer, The (I...|   4.706189155578613|
|  83439|Toys in the Attic...|   4.702956676483154|
+-------+--------------------+--------------------+
```

```
Command took 43.60 seconds -- by meghana.rwgsql@gmail.com at 8/22/2017, 5:46:31 PM on
Movies_Project
```