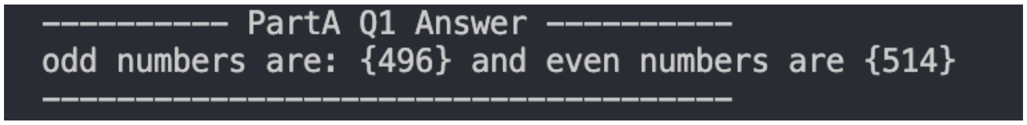
PART A Q1. Count the odd and even numbers using the file 'integer.txt' and download it from Quercus. Show your code and output.

Procedure: Load the text file into RDD format. Using filter to count the number of odd integers and the number of even integers.

Output shown as below.

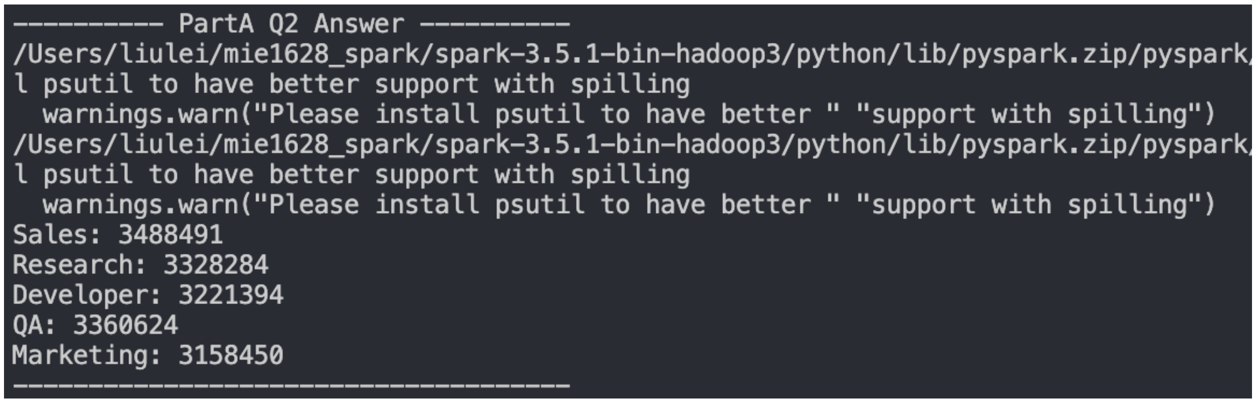


```
---------- PartA Q1 Answer ----------
odd numbers are: {496} and even numbers are {514}
-------------------------------------
```

PART A Q2. Calculate the salary sum per department using the file 'salary.txt' and download it from Quercus. Show the department name and salary sum. Show your code and output.

Procedure: Load the text file into RDD. Format RDD as key-value pairs where key stands the department in string, value as salary in integer. Reduce by key to calculate the sum of the value for each key.

Output shown as below.

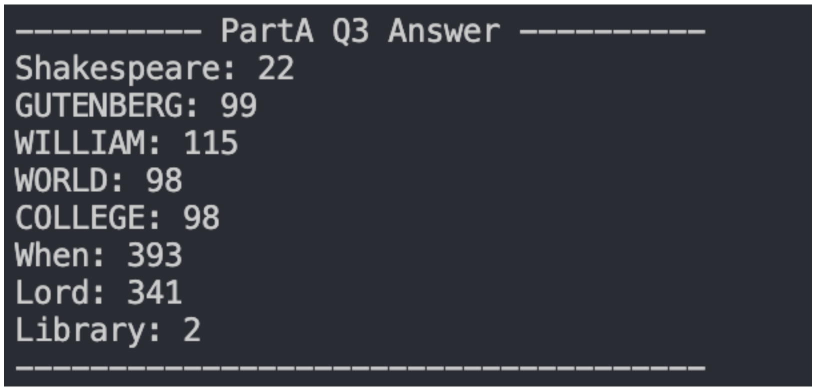


```
---------- PartA Q2 Answer ----------
/Users/liulei/mie1628_spark/spark-3.5.1-bin-hadoop3/python/lib/pyspark.zip/pyspark/
l psutil to have better support with spilling
  warnings.warn("Please install psutil to have better " "support with spilling")
/Users/liulei/mie1628_spark/spark-3.5.1-bin-hadoop3/python/lib/pyspark.zip/pyspark/
l psutil to have better support with spilling
  warnings.warn("Please install psutil to have better " "support with spilling")
Sales: 3488491
Research: 3328284
Developer: 3221394
QA: 3360624
Marketing: 3158450
-------------------------------------
```

PART A Q3. Implement MapReduce using Pyspark on file 'shakespeare.txt' and download it from the Quercus. Show how many times these particular words appear in the document: Shakespeare, When, Lord, Library, GUTENBERG, WILLIAM, COLLEGE and WORLD. (Count exact words only)

Procedure: Load the text file into RDD. Format RDD as key-value pairs where key stands for the word in Shakespeare.txt, and the value as the count of the word that appears in the text(using map and reducebykey). Then filter the result with the particular words specified in the problem.

Output shown as below.

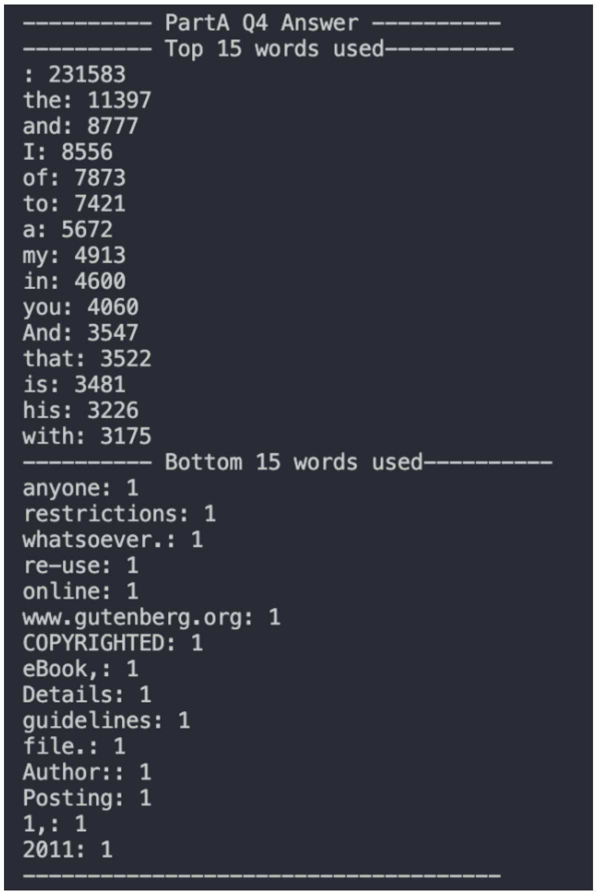


```
---------- PartA Q3 Answer ----------
Shakespeare: 22
GUTENBERG: 99
WILLIAM: 115
WORLD: 98
COLLEGE: 98
When: 393
Lord: 341
Library: 2
-------------------------------------
```

PART A Q4. Calculate the top 15 and bottom 15 words using the file “Shakespeare.txt” and download it from Quercus. Show 15 words with the most count and 15 words with the least count. You can limit by 15 in ascending and descending order of count. Show your code and output.

Procedure: From the key-value pairs (words-count)retrieved from Q3, use sortBy to determine the top 15 and bottom 15 words.

Output shown as below.

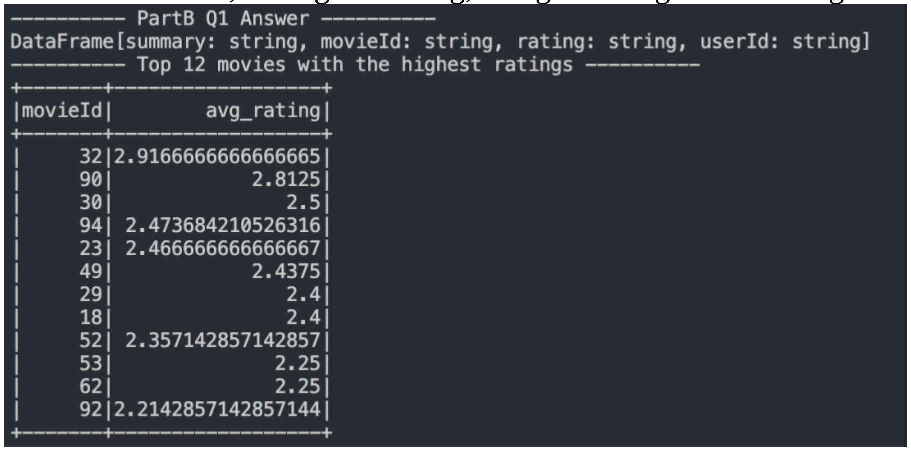


```
---------- PartA Q4 Answer ----------
---------- Top 15 words used----------
: 231583
the: 11397
and: 8777
I: 8556
of: 7873
to: 7421
a: 5672
my: 4913
in: 4600
you: 4060
And: 3547
that: 3522
is: 3481
his: 3226
with: 3175
---------- Bottom 15 words used----------
anyone: 1
restrictions: 1
whatsoever.: 1
re-use: 1
online: 1
www.gutenberg.org: 1
COPYRIGHTED: 1
eBook,: 1
Details: 1
guidelines: 1
file.: 1
Author:: 1
Posting: 1
1,: 1
2011: 1
-------------------------------------
```

PART B Q1. Describe your data. Calculate the top 12 movies with the highest ratings and the top 12 users who provided the highest ratings. Show your code and output.

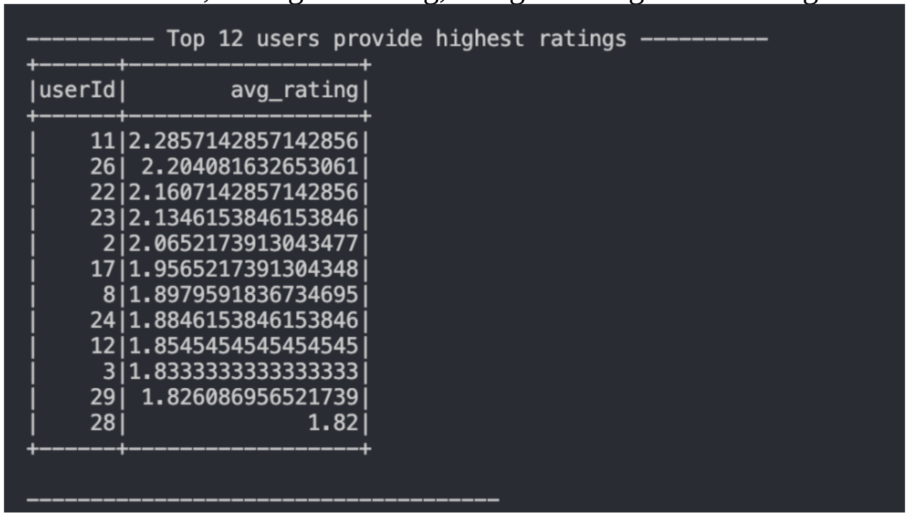
Procedure: Load csv into dataframe. Keep the header so we have column names: movieId, rating, and userId.

For each movieId, average its rating, and get the highest 12 ratings with its moveId.



```
---------- PartB Q1 Answer ----------
DataFrame[summary: string, movieId: string, rating: string, userId: string]
---------- Top 12 movies with the highest ratings ----------
+-------+------------------+
|movieId|        avg_rating|
+-------+------------------+
|     32|2.9166666666666665|
|     90|            2.8125|
|     30|               2.5|
|     94| 2.473684210526316|
|     23| 2.466666666666667|
|     49|            2.4375|
|     29|               2.4|
|     18|               2.4|
|     52| 2.357142857142857|
|     53|              2.25|
|     62|              2.25|
|     92|2.2142857142857144|
+-------+------------------+
```

For each userId, average its rating, and get the highest 12 ratings with itsuserId.

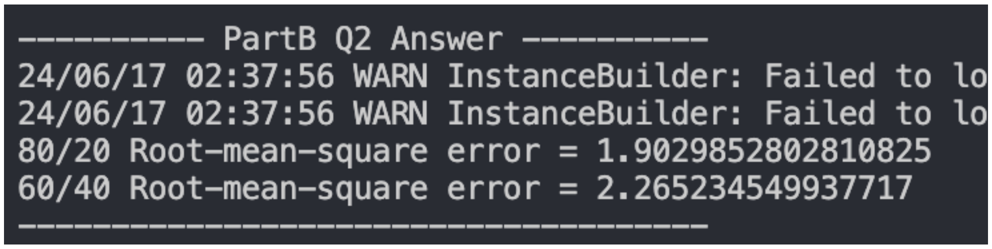


```
---------- Top 12 users provide highest ratings ----------
+------+------------------+
|userId|        avg_rating|
+------+------------------+
|    11|2.2857142857142856|
|    26| 2.204081632653061|
|    22|2.1607142857142856|
|    23|2.1346153846153846|
|     2|2.0652173913043477|
|    17|1.9565217391304348|
|     8|1.8979591836734695|
|    24|1.8846153846153846|
|    12|1.8545454545454545|
|     3|1.8333333333333333|
|    29| 1.826086956521739|
|    28|              1.82|
+------+------------------+

--------------------------------------
```

PART B Q2. Split the dataset into train and test. Try 2 different combinations for e.g. (60/40, 70/30, 75/25 and 80/20). (Train your model and use collaborative filtering approach on 70 percent of your data and test with the other 30 percent and so on). Show your code and output.

Procedure: I split the dataset into 80/20 and 60/40. After training, the rmse for two splits are as below.

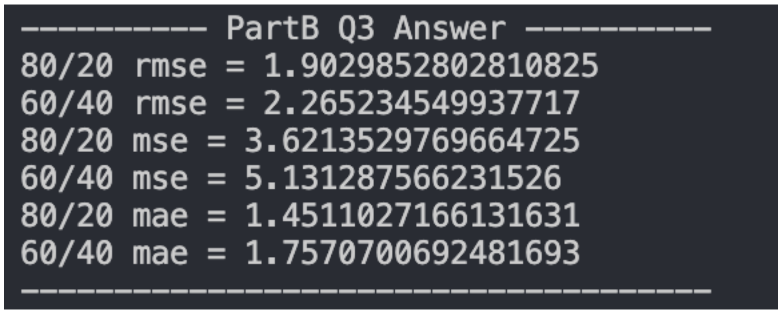


```
---------- PartB Q2 Answer ----------
24/06/17 02:37:56 WARN InstanceBuilder: Failed to lo
24/06/17 02:37:56 WARN InstanceBuilder: Failed to lo
80/20 Root-mean-square error = 1.9029852802810825
60/40 Root-mean-square error = 2.265234549937717
-------------------------------------
```

PART B Q3. Explain MSE, RMSE and MAE. Compare and evaluate both of your models with evaluation metrics (RMSE or MAE), show your code and print your results. Describe which one works better and why.

MSE measures the average squared difference between the estimated values and the actual value. RMSE is the square root of MSE. MAE measures the average of the absolute differences between predicted and actual values.

Since MAE is a pure liner measurement, it is less sensitive for outliners or input difference. As the results shows, MAE score the lowest error among the three evaluation metrics, but due to its less sensitive, the MAE evaluation looks less presentative. Mse score the highest, since the squared result makes it too sensitive with the data. As we don't want to heavily penalize the data difference, I prefer to use rmse for evaluation.



```
---------- PartB Q3 Answer ----------
80/20 rmse = 1.9029852802810825
60/40 rmse = 2.265234549937717
80/20 mse = 3.6213529769664725
60/40 mse = 5.131287566231526
80/20 mae = 1.4511027166131631
60/40 mae = 1.7570700692481693
-------------------------------------
```

PART B Q4. Now tune the parameters of your algorithm to get the best set of parameters. Explain different parameters of the algorithm which you have used for tuning your algorithm. Evaluate all your models again. Show your code with the best values and output.

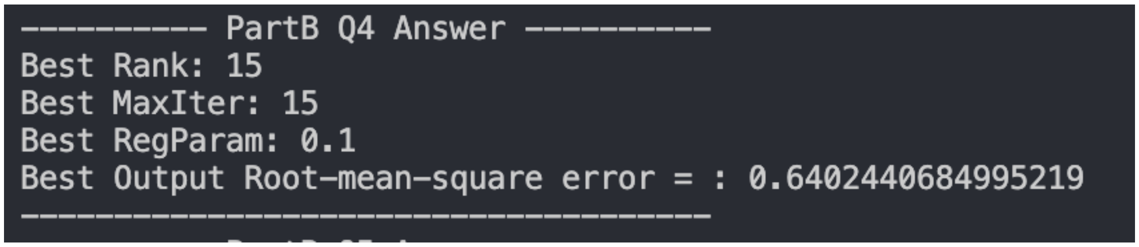
I tried tunning:

MaxIter: for different number of iterations to see if model get convergence and prevent it from overfitting.

RegParam: to help fits model's complexity and prevent it from overfitting or underfitting.

Rank: for adjust the number of latent factors that the model should take into consideration. It is also a way to adjust model's complexity and prevent it from overfitting or underfitting.

My best model's parameters with rmse are shown as below.



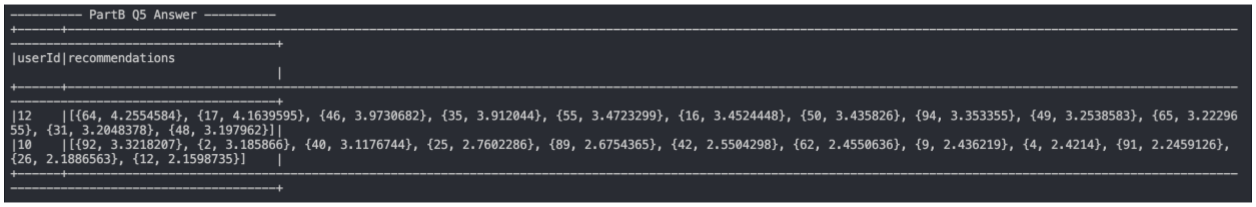
```
---------- PartB Q4 Answer ----------
Best Rank: 15
Best MaxIter: 15
Best RegParam: 0.1
Best Output Root-mean-square error = : 0.6402440684995219
-------------------------------------
```

PART B Q5. Calculate the top 12 movie recommendations for userID 10 and userID 12. Show your code and output.

Procedure: I used recommendForUserSubset to generate 12 recommendations for user 10 and user 12.

For user10, model recommends movies: 64, 17, 46, 35, 55, 16, 50, 94, 49, 65, 31, 48

For user12, model recommends movies: 92, 2, 40, 25, 89, 42, 62, 9, 4, 91, 26, 12



```
---------- PartB Q5 Answer ----------
+------+---------------------------------------------------------------------------------------------------------------------------------------
------------------------------------+
|userId|recommendations
                                    |
+------+---------------------------------------------------------------------------------------------------------------------------------------
------------------------------------+
|12    |[{64, 4.2554584}, {17, 4.1639595}, {46, 3.9730682}, {35, 3.912044}, {55, 3.4723299}, {16, 3.4524448}, {50, 3.435826}, {94, 3.353355}, {49, 3.2538583}, {65, 3.22296
55}, {31, 3.2048378}, {48, 3.197962}]|
|10    |[{92, 3.3218207}, {2, 3.185866}, {40, 3.1176744}, {25, 2.7602286}, {89, 2.6754365}, {42, 2.5504298}, {62, 2.4550636}, {9, 2.436219}, {4, 2.4214}, {91, 2.2459126},
{26, 2.1886563}, {12, 2.1598735}]    |
+------+---------------------------------------------------------------------------------------------------------------------------------------
------------------------------------+
```