Big Data

Master Program in Data Science & Analytics
Prof Flávio L Pinheiro and Prof Ian Scott

Six Degrees of Kevin Bacon

Introduction - Six Degrees of Kevin Bacon is a game based on the "six degrees of separation" concept, which posits that any two people on Earth are six or fewer acquaintance links apart. Movie buffs challenge each other to find the shortest path between an arbitrary actor and prolific actor Kevin Bacon. It rests on the assumption that anyone involved in the film industry can be linked through their film roles to Bacon within six steps.
The analysis of social networks can be a computationally intensive task, especially when dealing with large volumes of data. It is also a challenging problem to devise a correct methodology to infer an informative social network structure. Here, we will analyze a social network of actors and actresses that co-participated in movies. We will do some simple descriptive analysis, and in the end try to relate an actor/actress's position in the social network with the success of the movies in which they participate.

Rules & Notes - Please take your time to read the following points:

1. You have 36 hours to complete a list of tasks (page 2 to 4) that involve the load and parsing of data, network inference, and analysis;
2. The assignment starts on May 9th at noon, and ends at May 10th at midnight;
3. It is acceptable that you discuss with your colleagues different approaches to solve each step of the problem set, but the assignment is individual. That is, you are responsible for writing your own code, and analyzing the results. Clear cases of cheating will be penalized with 0 points in this assignment;
4. After review of your submission files, and before a mark is attributed, you might be called to orally defend your submission;
5. You will be scored first and foremost by the number of correct answers, secondly by the logic used in the trying to approach each step of the problem set;
6. You can perform the exercises in the platform you deem more appropriated for you (databricks or the Docker provided during the labs). However, we strongly recommend the databricks platform as the docker image we have been working with does not currently contain the graphFrames package;
7. Questions are divided into Easy (Green), Medium (Orange), and Hard (Red).
8. Solving correctly all Easy questions grants you 12 points;
9. Solving correctly all Easy and Medium questions grants you 18 points;
10. Solving correctly all questions grants you 20 points;
11. Consider skipping questions that you are stuck in, and get back to them later;
12. Expect computations to take a few minutes to finish in some of the steps.

Questions – Any **questions about this assignment** should be posted in the **Forum@Moodle**, questions by e-mail will not be answered. On **Friday (May 10th) from 10am to 12am** there will be an **open office** session in **Room 4** for anyone with questions concerning the assignment.

Delivery - To fulfill this activity you will have to upload the following materials to Moodle:

1. A PDF version of the question form with your answers;

2. A signed statement of authorship, which is present in the last page;
3. Files with all the code you have used to answer the questions, preferably a notebook (from databricks or Jupyter).

Data Sources and Description - We will use data from IMDB. You can download raw datafiles from https://datasets.imdbws.com. Note that the files are tab delimited (.tsv) You can find a description of the each datafile in https://www.imdb.com/interfaces/

Questions

1. Preliminary Analysis

a) Load the necessary data into Spark. Note that the data might require parsing and preprocessing to be ready for the follow up steps.

b) Using Spark Core or Spark SQL, write the necessary code to answer the following questions:

i) [EASY] Movies in the database can have multiple genres. Which combination of genres has the most votes?

ii) [EASY] Which combination of genres has the highest average IMDB rating, when including only titles with more than 500 votes?

iii) [MEDIUM] Which is the most common genre that is made (not combination of genres, this requires splitting cases where movies have multiple genres)?

iv) [HARD] What is the most common movie genre by year (2000-present) (by movie startYear) (as above this is by individual genre, not combination of genres)?

Hint on loading data into Data Bricks

1. You can use shell commands to download data from the internet straight to our cluster in databricks. For instance, the following command in a notebook cell:

```
%sh wget https://datasets.myData.com/myData.tsv.gz
```

Will download the file "myData.gz" into the databricks downloads data folder.

2. You can also use the following command:

```
%sh gunzip title.myData.tsv.gz
```

to extract compressed files (in this case a .gz file) into the the databricks downloads data folder. In this case you will find the extracted file located in

```
file:/databricks/driver/title.myData.tsv
```

3. Finally, you can load the file into a spark DataFrame with the command:

```
df = spark.read.option("sep","\t")
          .option("header","true")
          .option("inferSchema","true")
          .csv("file:/databricks/driver/myData.tsv")
```

2. Network Inference, Let's build a network

a) We want to examine a network that abstracts how actors and actress are related through their co-participation in movies. To that end perform the following steps:

i) [EASY] Create an RDD or DataFrame that combines titles (i.e., movies, tv-shows, etc …) and the participants (i.e., actors, directors, etc …);

ii) [EASY] Remove from the DataFrame or RDD obtained in the previous step: Any participant that is not an actor or actress; All adult movies; All dead actors or actresses; All actors or actresses born before 1920 or with no date of birth listed; All titles that are not of the type movie.

iii) [EASY] How many actors does your final RDD/DataFrame contains?

iv) [MEDIUM] Generate an RDD or DataFrame that lists all links of your network. Here we shall consider that a link connects a pair of actors if they participated in at least one movie together. For every link we then need anytime a pair of actors were together in a movie as a link in each direction (A -> B and B -> A). However links should be distinct we do not need duplicates when two actors worked together in several movies;

b) Compute the page rank of each actor. This can be done using [MEDIUM] GraphFrames or by using [HARD] RDDs and iterative implementation of the PageRank algorithm. Do not take more than 5 iterations in either case otherwise it will take too long to solve. [EASY] List the top 10 and bottom 10 actors by their page ranks.

ii) [EASY] What is the average page rank of movie actors by birth year?

c) [EASY] Compute the degree (i.e., number of links) of each actor/actress and list the top 10 and bottom 10 actors by their degrees.

3. Let's play Kevin's own game

a) Using Spark GraphFrame and/or Spark Core library perform the following steps:

i) [EASY] Identify the id of Kevin Bacon, there are two actors named 'Kevin Bacon', we will use the one with the highest degree, that is, the one that participated in most titles;

ii) [MEDIUM] Estimate the shortest path between every actor in the database actors and Kevin Bacon;

iii) [MEDIUM] Summarise the data, that is, count the number of actors at each number of degress from kevin bacon (you will need to deal with actors unconnected to kevin bacon).

Hint on GraphFrames on Databricks

If you have trouble with graphframes in databricks (specifically the import statement) you need to make sure the graphframes package is installed on the cluster you are running. If you click home on the left, then click on the graphframes library which you loaded in Lab 11 you can install the package on your cluster (check the graphframes checkbox and click install).

Hint on Shortest Paths

A path in a network is measured by the minimum number of links that connect any two pairs of nodes in the network. Check the GraphFrame documentation, Spark already comes with a method to estimate the length of the shortest path between any two arbitrary nodes.

Hint on PageRank

Page Rank is a centrality algorithm initially proposed by the founders of Google to score webpages. It has since then been applied in many other applications of Network Analysis. You can find more information at http://www.openkb.info/2016/03/understanding-pagerank-algorithm-in.html (example implementation in Spark Scala) or by Ian Rogers at https://www.cs.princeton.edu/~chazelle/courses/BIB/pagerank.htm . Moreover, you already implemented this algorithm in Lab **11**.

4. Engineering the perfect cast

a) Using spark Core or SQL and spark ML perform the following steps using the data you filtered in step 2.ii:

i) [EASY] Create a DataFrame with the IMDB rating of each movie;

ii) [EASY] Create a DataFrame with the average pagerank of the cast in each movie (Note: for 4.a.ii you can use the results previously computed in 2.b., if you were unable to calculate the pagerank use the average age of the actors at the time of the movie release)

iii) [EASY] Create a DataFrame with the average degree of the cast in each movie (Note: for 4.a.iii you case use the results previously computed in 2.c, if you were unable to calculate the degrees you can use the average number of titles each actor starred in))

iv) [MEDIUM] Using spark ML, estimate a linear model that regresses the IMDB rating of a movie as a function of the average pagerank and average the degree.

vi) [MEDIUM] Print the coefficients of the regression