**DEVELOPER GUIDE**

**Analyzing Trust Relationships to Facilitate Scientific Service Discovery and Recommendation**

By

Team NASA, CMU SV Practicum Fall 2013

Venkatesh Sriram- [s.venkatesh.5789@gmail.com](mailto:venkatesh.sriram@sv.cmu.edu)

Shuai Wang- [cmu.swang@gmail.com](mailto:cmu.swang@gmail.com)

Neeraj Saini- [sainieneeraj@gmail.com](mailto:sainieneeraj@gmail.com)

Pujita Rao- [rao.pujita@gmail.com](mailto:rao.pujita@gmail.com)

Faculty advisor:

Dr. Jia Zhang

Associate Research Professor

Carnegie Mellon University  
Silicon Valley

[jia.zhang@sv.cmu.edu](mailto:jia.zhang@sv.cmu.edu)

Table of Contents

GETTING RESULTS IN REST API USING JAVA PLAY! 2

VISUALIZING CO-AUTHOR GRAPHS 3

MACHINE LEARNING AND DATABASE MANAGEMENT 6

1)CSV FILE PREPARATION- 6

2)DATA PREPARATION 7

3)RUNNING MACHINE LEARNING ALGORITHMS 10

# 

# GETTING RESULTS IN REST API USING JAVA PLAY!

1. First, you need to install Java Play!. This is a very simple process, and you just need to do what is said in <http://www.playframework.com/documentation/2.0/Installing>
2. In the terminal, navigate to the Project source folder. Then type the following-

*play run*

This sets up the server and your play app is ready to go.

1. The following are the URL’s you need to type in your browser

**NOTE - Unless you host it, it should be after localhost:9000 (e.g. localhost:9000/getReputation/Otto Muzik)**

a) Get a graph of the co-authors of an author given a level and a name, returns in JSON format

/getGraphWithoutRender/:name/:level

b) Same as getGraphWithoutRender except that it also calls the JUNG framework to render the graph, if you have it installed

/getGraphWithRender/:name/:level

c) Given an author's name, get all the co-authors, JSON Format

/getCoAuthorInformation/:name

d) Given an author's name (part of the dataset), it returns the trust value of the author

/getReputation/:name

e) Given an author's name (part of the dataset), and a topic, it returns the trust value of the author for that topic

/getReputationForTopic/:name/:topic

f) Given an author's name (part of the dataset), it returns the entire co-author graph beginning with that author

/getSocialNetwork/:name

g) Given an author's name (part of the dataset), and a topic, it returns all the co-authors for that topic

/getCoAuthorsByTopic/:name/:topics

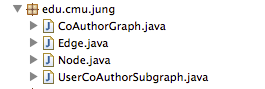
h) Given an author's name (part of the dataset), a topic, and a year, it returns all the co-authors for that topic after that year

/getCoAuthorsByTopicAndTime/:name/:topics/:year

1. In case you want to host it on Heroku, the following page will help <https://blog.heroku.com/archives/2011/8/29/play>

# VISUALIZING CO-AUTHOR GRAPHS

The package called *edu.cmu.jung* is the package that contains all the files related to creating and visualizing the co-author graph.

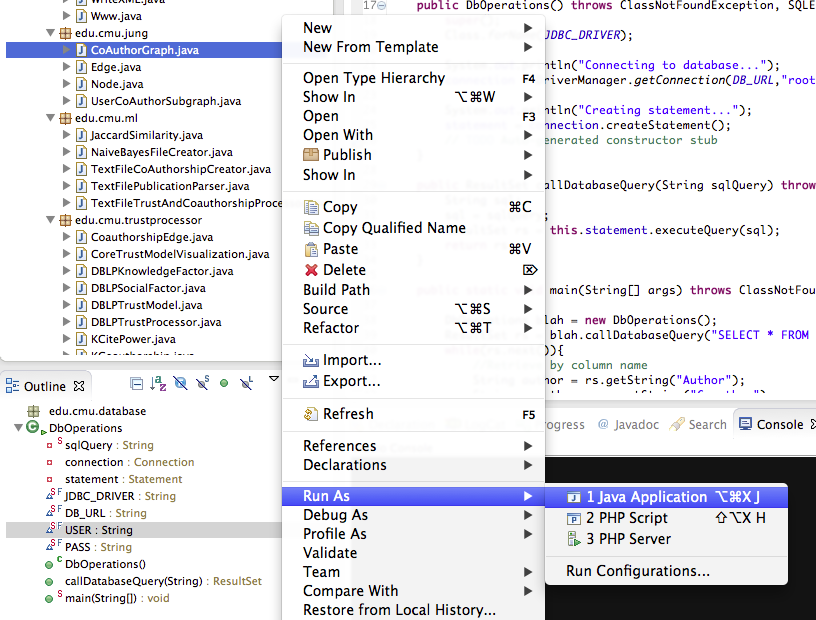


To visualize the entire co-author graph

Ensure that the project folder contains the XML file based on which you want to generate the co-author graph.

The steps you need to take are-

1. Go to *edu.cmu.jung*
2. Run the CoAuthorGraph.java file

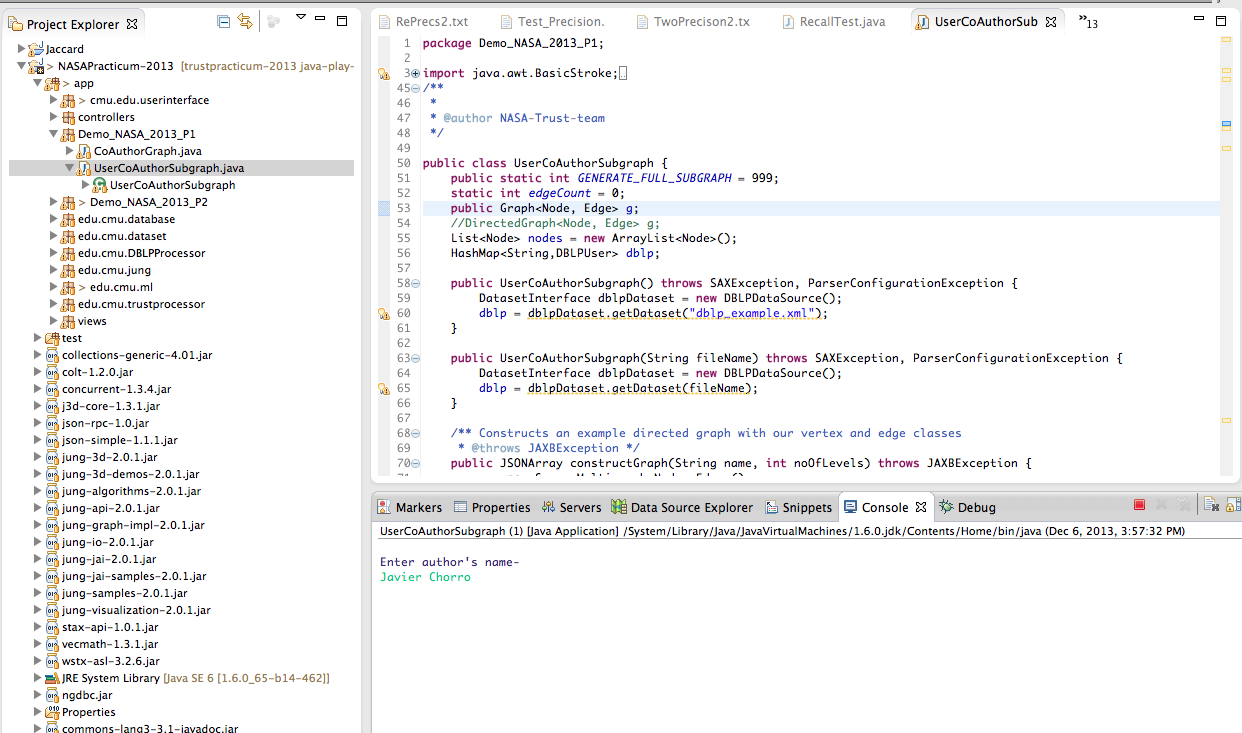


3) You get the output as both a graph, and a JSON representation in the console. This graph is based on *dblp\_example.xml*

The steps you need to take are-

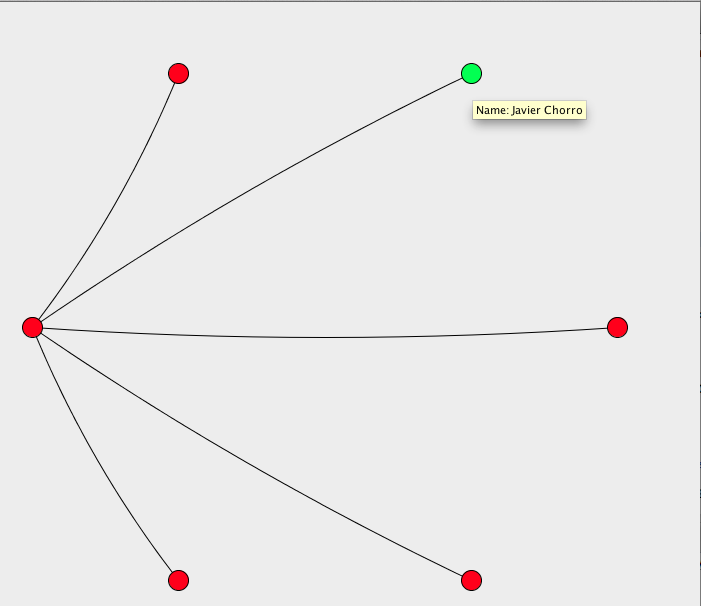
1) Go to *Demo\_NASA\_2013\_P1*

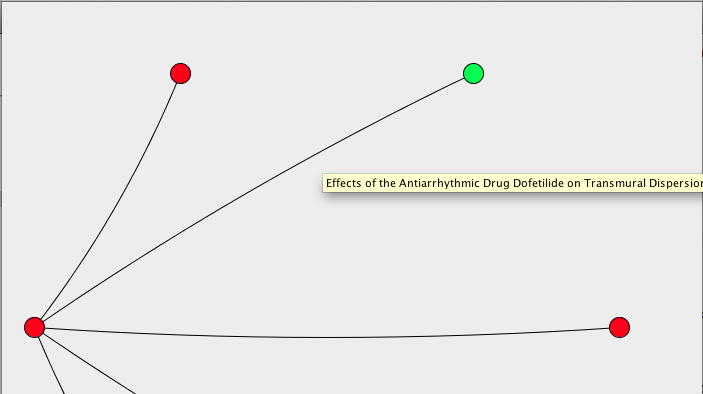
2) Run the UserCoAuthorSubGraph.java file, in the console put the author name you are interested.



3) You get the output as both a graph, and a JSON representation in the console. This graph is based on *dblp\_example.xml.*

Output screen shots:





4) If you hover over your mouse to one of the nodes it shows the author name and the edge will show the list of their publications.

# MACHINE LEARNING AND DATABASE MANAGEMENT

## CSV FILE PREPARATION-

1. Put dblp.xml in your project directory.
2. Run splitfiles.py on dblp.xml python splitfiles.py dblp.xml

This splits dblp.xml into 100 files named <split\_dblp\_40.xml> which makes it manageable to parse using JAXB.

1. Go to edu.cmu.DBLPProcessor package
2. Run FullDBLPDataToTextConverter.java



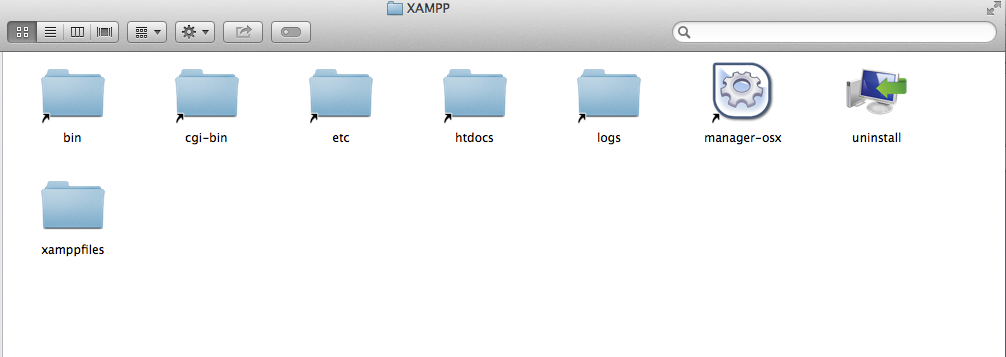
Here, i represents the file range on which to conduct the operations (i.e. split\_dblp\_i.xml).

1. Run the program 5 times, each time changing the range of i in sets of 20 (i.e. 1-20, 21-40… 81-100). This is because the Java Heap runs out if you do all 100 at the same time.
2. This puts in all the data in a file called dblpdata.txt.
3. To get the lines relevant to the topics that you want, make the required changes to *getrelevantlines.py* and run that python script on *dblpdata.txt.*

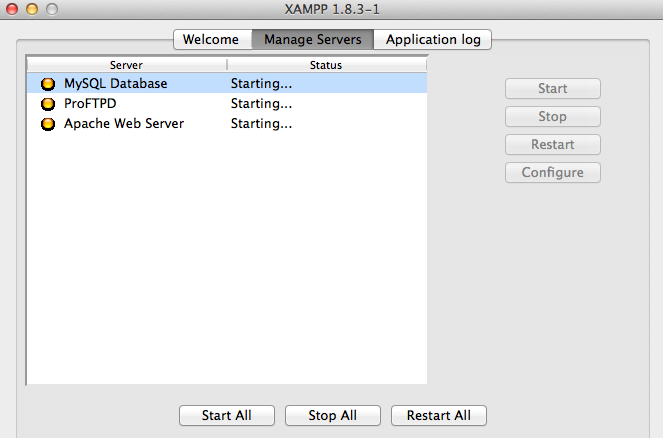
***NOTE:- YOU SHOULD NOT NEED TO DO THIS. THE RELEVANT CSV FILE IS IN dblp.csv IN PROJECT DIRECTORY***

## DATA PREPARATION

* To get the DBLP and retrieve data dynamically, we are using mysql database. Download link: <http://dev.mysql.com/downloads/>
* Also click on build configuration to add mysql-connector-java-5.1.27-bin.jar file. It is under the folder after you download mysql
* Also download MySql workbench for database management, each time you want to run the code, make sure it is up running, I am using XAMPP library for mac, link:http://sourceforge.net/projects/xampp/files/XAMPP%20Mac%20OS%20X/ the following shows the library content:



* Click on manager—osx, and click on Mysql database will make sure that it is started. Also Mysql Workbench can allow you to check database status too.

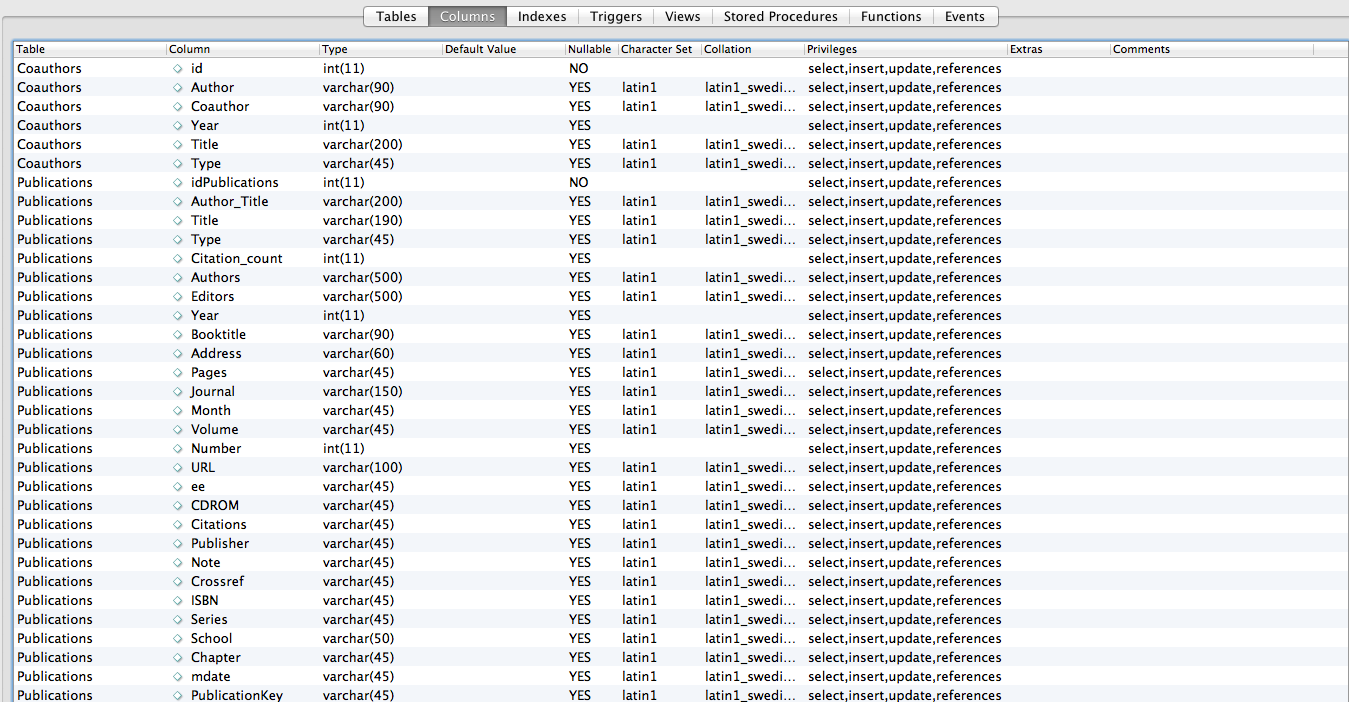


* When you have downloaded everything, the icon looks like:



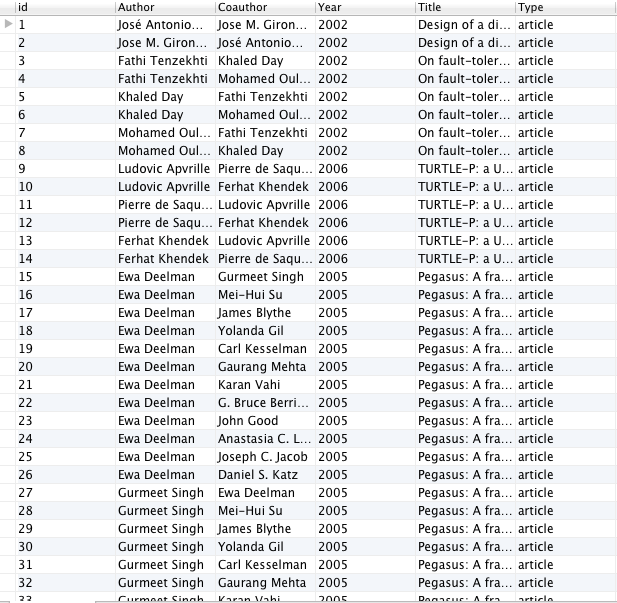
* Open the workbench, we need to construct the database schema first, then two database tables will be used later, including Coauthors and Publications

Table column screen shot:





* Look at CSV FILE PREPARATION. After getting the csv files, we can click on Import button in the middle and import the data manually. 
* Open the workbench, part of the data in table coauthors is shown as follows:

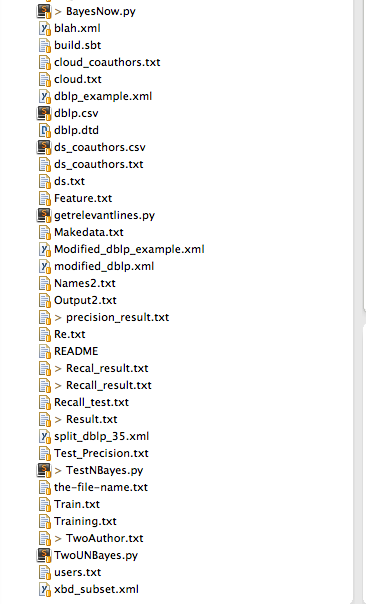


* Part of the data in Publications table is shown as follows:



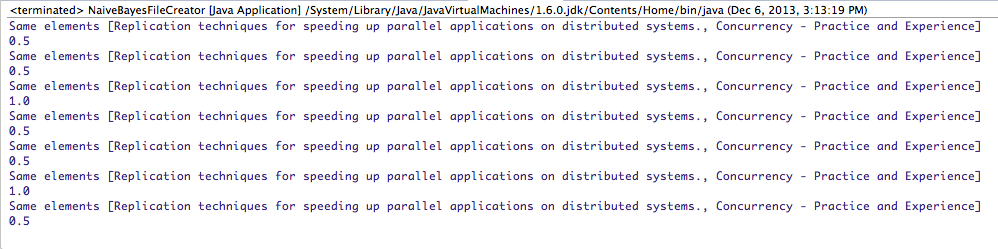
* After database configuration, we can open the project folder in eclipse:

Make sure that BayesNow.py, TestNBayes and TwoUNBayes.py are under the project folder,

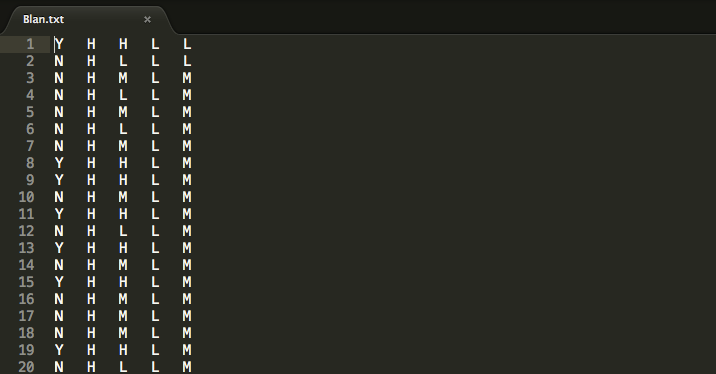


Generate training data and test data:

* Run the code NaiveBayesFileCreator under edu.cmu.ml package, it should print out sth like this, u can specify the file path in the main function second parameter.



* If you open the the output file, it will show sth like this:



* Then you can randomly choose part of the records (10% recommendated) as test file, the other 90% as training file.

Generate feature file:

* Create a file called Feature.txt, the content should be like this:

Feature\_Name {Set\_of\_Values}

domain-similarity{ H, M, L}

reputation-similarity { H, M, L}

connectedness-similarity { H, M, L}

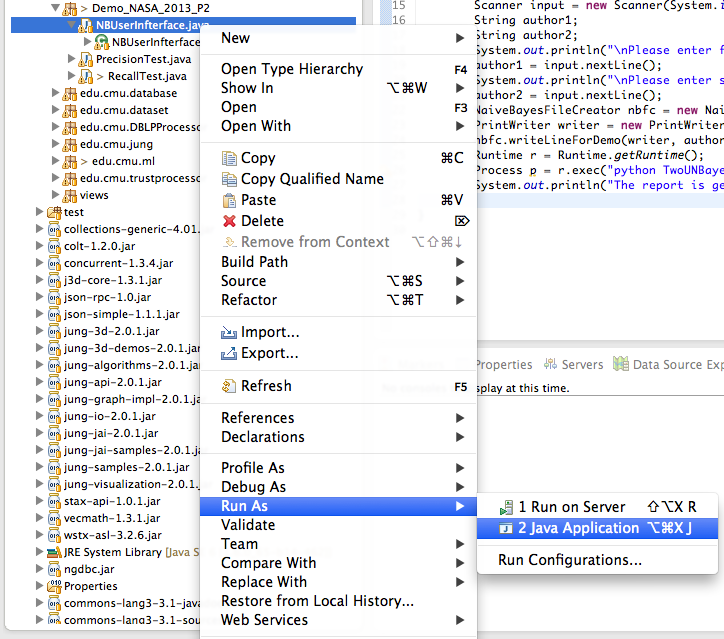
coauthorship-history{H,M,L}

Anytime you want to add more feature just add another line

## RUNNING MACHINE LEARNING ALGORITHMS

**Show probability of two specific authors to collaborate:**

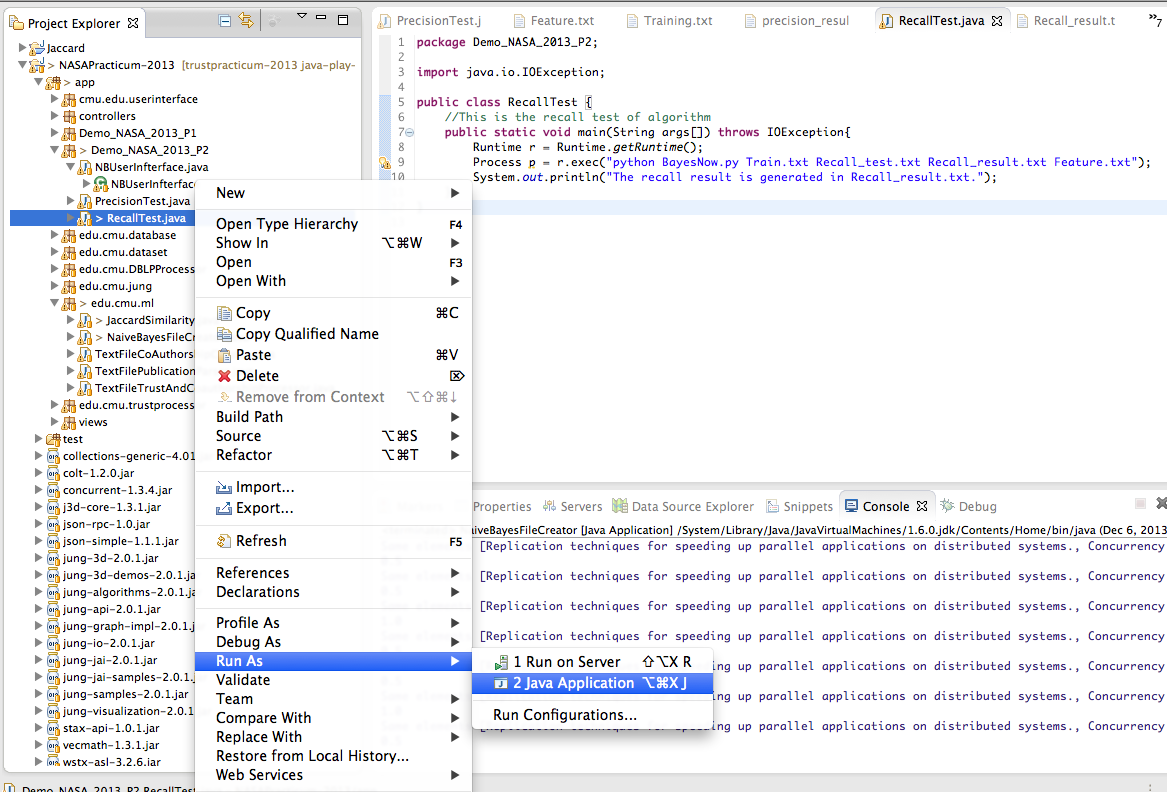
* Click on Demo\_NASA\_2013\_P2 folder, click on run as java application,



Then you will input two authors name as input to see their collaboration probability and also the detailed the feature information.

**Precision test and recall test:**

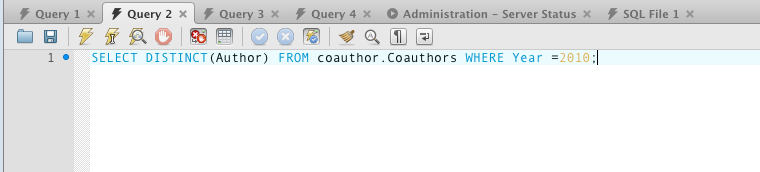
You can simply run the RecallTest.java file like this:



Also u can use any data you created as input, the second parameter will be the training file and third one are the test files. Finally you can see a report generated in the project folder called Recall\_result.txt. (Which you can easily rename as method input)

For precision test:

We need to have the real co-authorship information in real life (any co-authorship after 2010), to do that, you need to run a SQL Query in Mysql Workbench:



Then click on export data icon to one csv file. I am naming it Au2020.csv.

Later see the PrecisionTest.java line 29, put the name of your file there as input parameter like:

br=newBufferedReader(newFileReader("/Users/ShuaiWang/Desktop/Au2010.csv"));

Then you can run the code, it will actually take a while but the output should give you a number, in my experiment it shows: 

Then you can refer to the precision\_result.txt, it will show detail of each input, scroll down to the end of the file it shows:

