Social Network Analysis of YouTube.com

Project Report

Submitted By:

Nitin Kathpalia(15BCE0866)

Saloni Srivastava(15BCE2033)

Raghav Mahajan(15BCE0355)

Prepared for:

Social and Information Networks(CSE3021)-Project Component

Submitted to:

Prof. Annapurna J.

VIT®
UNIVERSITY
(Estd. u/s 3 of UGC Act 1956)
VELLORE ▪ CHENNAI
www.vit.ac.in

SCOPE

Table of Contents:

1. Introduction

The Internet has witnessed an explosion of networked video sharing as a new killer application in the recent two years. Among them, YouTube is the most successful one, with more than 100 million videos being watched every day. The expensive deal by Google, as well as the success of similar sites (i.e., Tudou) further confirm the mass market interest. Their great achievement lies in the combination of rich media and, more importantly, their social networks. These sites have created a video community on the web, where anyone can be a star, from lip-synching teenage girls to skateboarding dogs. With no doubt, they are reshaping popular culture and the way people surf the Internet.

Since its establishment in early 2005, YouTube has become one of the fastestgrowing websites, and ranks second in traffic among all the websites in the Internet by the survey of Alexa. It has a significant impact on the Internet traffic, but itself is suffering from severe scalability constraints.

By now, we have obtained a dataset containing of about 8 attributes with around 1 million users and videos.

More importantly, we also investigate the social networking aspect of YouTube, as this is the most unique part and a key driving force towards the success of YouTube and similar sites. In particular, we have found that the graph of YouTube's videos' structure has a clear small-world characteristic. This indicates that the videos have strong correlations with each other, and creates opportunities for developing novel techniques to enhance its service quality.

2. Problem Statement

According to Alexa, the current speed of YouTube is “Slow” (average load time is 3.6 seconds) and is slower than 69% of the surveyed sites. Therefore, understanding the features of YouTube and similar video sharing sites is essential to network traffic engineering and to their sustainable enhancement.

3. Abstract

YouTube has become the most successful Internet website providing a new generation of short video sharing service since its establishment in early 2005. YouTube has a great impact on Internet traffic nowadays, yet itself is suffering from a severe problem of scalability. Therefore, understanding the characteristics of YouTube and similar sites is essential to network traffic engineering and to their sustainable development. To this end, we have crawled the YouTube site for this month, collecting about 1 million YouTube videos' and users’ data. In this project, we will present a systematic and in-depth measurement study on the statistics of YouTube videos. We have found that YouTube videos have noticeably different statistics compared to traditional streaming videos, ranging from length and access pattern, to their growth trend and active life span. We investigate the social networking in YouTube videos, as this is a key driving force toward its success. We find that the links to related videos generated by uploaders' choices have clear small-world characteristics. This indicates that the videos have strong correlations with each other, and creates opportunities for developing novel techniques to enhance the service quality.

We will use the collected datasets and their attributes to present a statistical analysis by calculating the centrality measures, user behavior, their interconnectivity, trend in videos, their correlation etc. All these will be represented in the form of graphs and give a complete overlook if the analysis done using obtained datasets.

4. Literature Survey

The exploration led so far for question discovery and following articles in user ratings based recommendation systems. The arrangement of difficulties plot above traverse a few areas of research and the larger part of important work. A following framework ought to have the capacity to foresee the position of any impeded articles.

- **“Characterizing User Access To Videos On The World Wide Web”** by Soam Acharya, Brian Smith and Peter Parnes. this paper presents an analysis of trace data obtained from an ongoing VOW experiment in Luleå University of Technology, Sweden. This experiment is unique as video material is distributed over a high bandwidth network allowing users to make access decisions without the network being a major factor.
- **“Deep Neural Networks for YouTube Recommendations”** by Paul Covington, Jay Adams, Emre Sargin. This paper focusses on the immense impact deep learning has recently had on the YouTube video recommendations system.
- **“Long-term Streaming Media Server Workload Analysis and Modeling”** by Wenting Tang, Yun Fu, Ludmila Cherkasova, Amin Vahdat. Currently, Internet hosting centers and content distribution networks leverage statistical multiplexing to meet the performance requirements of a number of

competing hosted network services. Developing efficient resource allocation mechanisms for such services requires an understanding of both the short-term and long-term behavior of client access patterns to these competing services.

- **"Influence in Ratings-Based Recommender Systems: An Algorithm-Independent Approach"** by Al Mamunur Rashid George Karypis John Riedl. Influence is a measure of the effect of a user on the recommendations from a recommender system. Influence is a powerful tool for understanding the workings of a recommender system. Experiments show that users have widely varying degrees of influence in ratings-based recommender systems.
- **"Recommendation algorithm based on item quality and user rating preferences"** by Yuan GUAN, Shimin CAI, Mingsheng SHANG. We propose a new recommendation algorithm based on item quality and user rating preferences, which can significantly decrease the computing complexity. Besides, it is interpretable and works better when the data is sparse. Through extensive experiments on three benchmark data sets, we show that our algorithm achieves higher accuracy in rating prediction compared with the traditional approaches.
- **"Social network analysis of the video bloggers' community in YouTube"** by Anusha Mogallapu. This research studied the structure of the social network of the video blogger community on YouTube. It analyzed the social network structure of friends and subscribers of the 187 video bloggers on YouTube and calculated the social network measures. This thesis compares the results to the structure described by Warmbrodt et al. in 2007 and explains the reasons for the distinctions.
- **"Web based Recommender Systems and Rating Prediction"** by Tho Nguyen. This project implements a recommender system on large dataset of Netflix's movies. This project also tries to improve recommender systems by incorporating confidence interval and genres of movies. This new approach enhances the performance and quality of service of recommender systems and gives better result than Netflix commercial recommender system, Cinematch.

5. Methodologies

The various phases of the project are-

i) Obtaining Data sets
ii) YouTube videos statistics
iii) User Statistics
iv) Statistical analysis of these attributes

v) Identifying patterns

A. Obtaining Data sets

The first step for the project was to obtain the appropriate datasets of YouTube about the videos and the users. We obtained these datasets from an online source who used crawlers during the months of January-March 2017 to mine data about videos and users from the site.

B. YouTube video statistics

The data includes video attributes such as

i) Uploader
ii) Video length
iii) Video size
iv) Date added

We'll analyze these and obtain useful patterns from the data sets mined.

C. User statistics

The data includes user statistics such as

i) User id
ii) Relation/connection with other users
iii) Location

6. Implementation

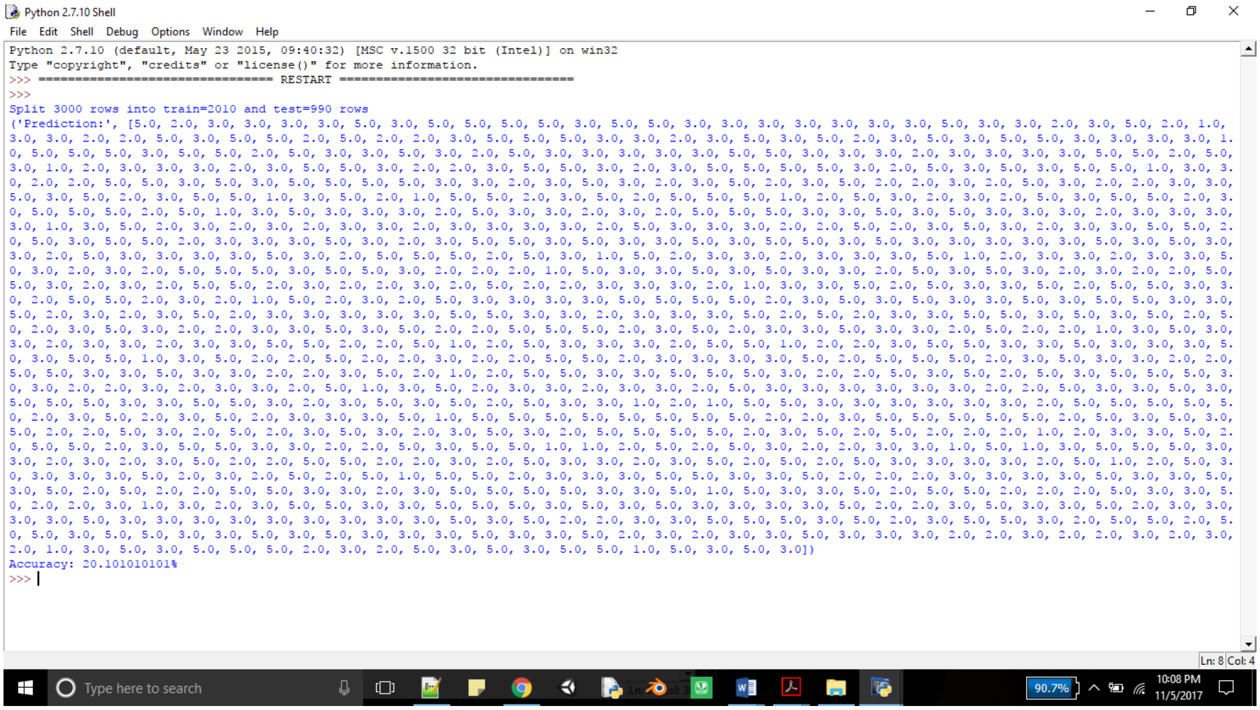
To provide better recommendations to YouTube users the algorithm uses the subscriptions of the user. If a subscribed channel rates another channel highly. Also, the algorithm predicts the ratings a channel might give to another channel and if these predicted ratings are high as well as the subscribed channel has rated it highly that channel will be recommended to the user.

Code used:

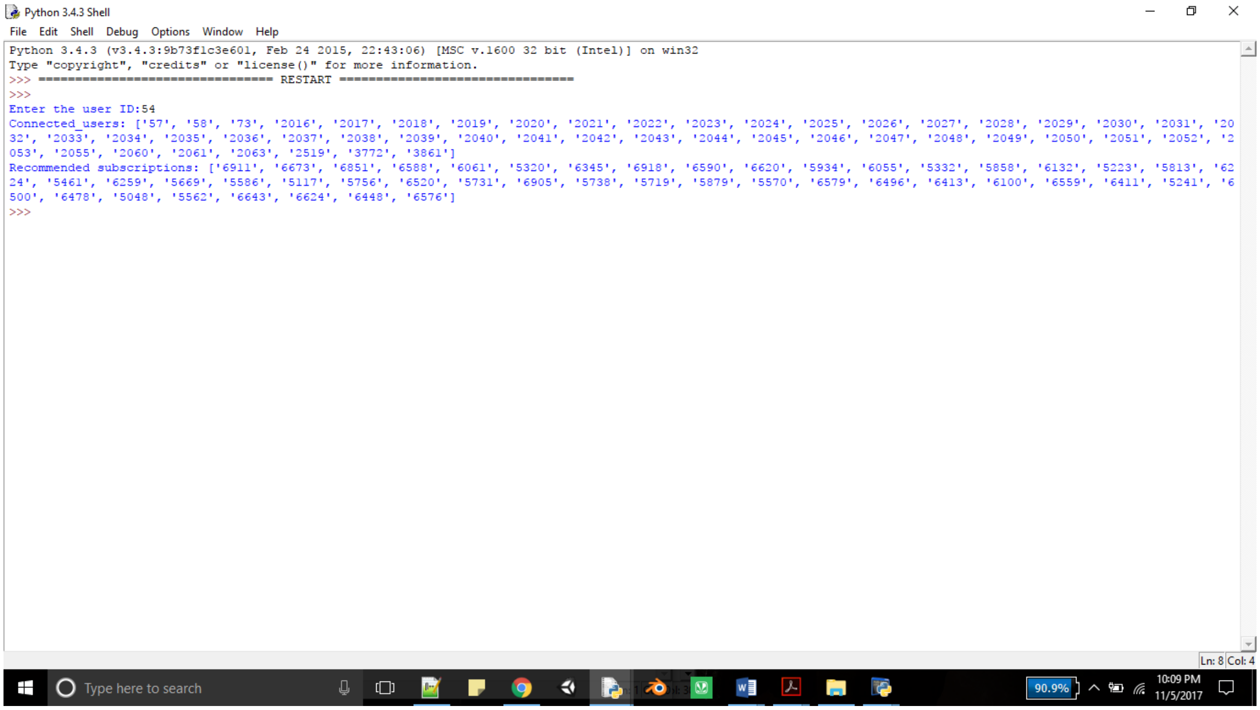
```
import csv
import random
import math

def loadCsv(filename):
lines = csv.reader(open(filename, "rb"))
dataset = list(lines)
for i in range(len(dataset)):
    dataset[i] = [float(x) for x in dataset[i]]
return dataset

def splitDataset(dataset, splitRatio):
trainSize = int(len(dataset) * splitRatio)
trainSet = []
copy = list(dataset)
while len(trainSet) < trainSize:
    index = random.randrange(len(copy))
    trainSet.append(copy.pop(index))
return [trainSet, copy]

def separateByClass(dataset):
separated = {}
for i in range(len(dataset)):
    vector = dataset[i]
    if (vector[-1] not in separated):
            separated[vector[-1]] = []
    separated[vector[-1]].append(vector)
return separated

def mean(numbers):
return sum(numbers)/float(len(numbers))

def stdev(numbers):
avg = mean(numbers)
variance = sum([pow(x-avg,2) for x in
numbers])/float(len(numbers)-1)
return math.sqrt(variance)

def summarize(dataset):
summaries = [(mean(attribute),
stdev(attribute)) for attribute in
zip(*dataset)]
del summaries[-1]
return summaries

def summarizeByClass(dataset):
separated = separateByClass(dataset)
summaries = {}
```

```
for classValue, instances in
separated.iteritems():
    summaries[classValue] =
summarize(instances)
return summaries

def calculateProbability(x, mean, stdev):
exponent = math.exp(-(math.pow(x-
mean,2)/(2*math.pow(stdev,2))))
return (1 / (math.sqrt(2*math.pi) * stdev))
* exponent

def calculateClassProbabilities(summaries,
inputVector):
probabilities = {}
for classValue, classSummaries in
summaries.iteritems():
    probabilities[classValue] = 1
    for i in range(len(classSummaries)):
            mean, stdev = classSummaries[i]
            x = inputVector[i]
            probabilities[classValue] *=
calculateProbability(x, mean, stdev)
return probabilities

def predict(summaries, inputVector):
probabilities =
calculateClassProbabilities(summaries,
inputVector)
bestLabel, bestProb = None, -1
for classValue, probability in
probabilities.iteritems():
    if bestLabel is None or probability >
bestProb:
            bestProb = probability
            bestLabel = classValue
return bestLabel

def getPredictions(summaries, testSet):
predictions = []
for i in range(len(testSet)):
    result = predict(summaries, testSet[i])
    predictions.append(result)
return predictions

def getAccuracy(testSet, predictions):
correct = 0
for i in range(len(testSet)):
    if testSet[i][-1] == predictions[i]:
            correct += 1
return (correct/float(len(testSet))) * 100.0

def main():
filename = 'ratings1.csv'
splitRatio = 0.67
dataset = loadCsv(filename)
trainingSet, testSet = splitDataset(dataset,
splitRatio)
print('Split {0} rows into train={1} and
test={2} rows').format(len(dataset),
len(trainingSet), len(testSet))
# prepare model
summaries =
summarizeByClass(trainingSet)
# test model
```

```
predictions = getPredictions(summaries,
testSet)
accuracy = getAccuracy(testSet,
predictions)
print('Prediction:',predictions)
print('Accuracy: {0}%').format(accuracy)

main()
```



```
Python 2.7.10 (default, May 23 2015, 09:40:32) [MSC v.1500 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>> 
Split 3000 rows into train=2010 and test=990 rows
('Prediction:', [5.0, 2.0, 3.0, 3.0, 3.0, 3.0, 5.0, 3.0, 5.0, 5.0, 5.0, 5.0, 3.0, 5.0, 5.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 5.0, 3.0, 3.0, 2.0, 3.0, 5.0, 2.0, 1.0,
3.0, 3.0, 2.0, 2.0, 5.0, 3.0, 5.0, 5.0, 2.0, 5.0, 2.0, 2.0, 3.0, 5.0, 5.0, 5.0, 3.0, 3.0, 2.0, 3.0, 5.0, 3.0, 5.0, 2.0, 3.0, 5.0, 3.0, 5.0, 5.0, 3.0, 3.0, 3.0, 3.0, 1.
0, 5.0, 5.0, 5.0, 3.0, 5.0, 5.0, 2.0, 5.0, 3.0, 3.0, 5.0, 3.0, 2.0, 5.0, 3.0, 3.0, 3.0, 3.0, 3.0, 5.0, 5.0, 3.0, 3.0, 3.0, 2.0, 3.0, 3.0, 3.0, 3.0, 5.0, 5.0, 2.0, 5.0,
3.0, 1.0, 2.0, 3.0, 3.0, 3.0, 2.0, 3.0, 5.0, 5.0, 3.0, 2.0, 2.0, 3.0, 5.0, 5.0, 3.0, 2.0, 3.0, 5.0, 5.0, 5.0, 5.0, 3.0, 2.0, 5.0, 3.0, 5.0, 3.0, 5.0, 5.0, 1.0, 3.0, 3.
0, 2.0, 2.0, 5.0, 5.0, 3.0, 5.0, 3.0, 5.0, 5.0, 5.0, 5.0, 3.0, 3.0, 2.0, 3.0, 5.0, 3.0, 2.0, 3.0, 5.0, 2.0, 3.0, 5.0, 2.0, 2.0, 3.0, 2.0, 5.0, 3.0, 2.0, 2.0, 3.0, 3.0,
5.0, 3.0, 5.0, 2.0, 3.0, 5.0, 5.0, 1.0, 3.0, 5.0, 2.0, 1.0, 5.0, 5.0, 2.0, 3.0, 5.0, 2.0, 5.0, 5.0, 5.0, 1.0, 2.0, 5.0, 3.0, 2.0, 3.0, 2.0, 5.0, 3.0, 5.0, 5.0, 2.0, 3.
0, 5.0, 5.0, 5.0, 2.0, 5.0, 1.0, 3.0, 5.0, 3.0, 3.0, 3.0, 2.0, 5.0, 3.0, 3.0, 2.0, 3.0, 2.0, 5.0, 5.0, 5.0, 3.0, 3.0, 5.0, 3.0, 5.0, 3.0, 3.0, 3.0, 2.0, 3.0, 3.0, 3.0,
3.0, 1.0, 3.0, 5.0, 2.0, 3.0, 2.0, 3.0, 2.0, 3.0, 3.0, 2.0, 3.0, 3.0, 3.0, 3.0, 2.0, 5.0, 3.0, 3.0, 3.0, 2.0, 2.0, 5.0, 2.0, 3.0, 5.0, 3.0, 2.0, 3.0, 3.0, 5.0, 5.0, 2.
0, 5.0, 3.0, 5.0, 5.0, 2.0, 3.0, 3.0, 3.0, 5.0, 3.0, 2.0, 3.0, 5.0, 5.0, 3.0, 5.0, 3.0, 3.0, 5.0, 3.0, 5.0, 5.0, 3.0, 5.0, 3.0, 3.0, 3.0, 3.0, 3.0, 5.0, 3.0, 5.0, 3.0,
3.0, 2.0, 5.0, 3.0, 3.0, 3.0, 3.0, 5.0, 3.0, 2.0, 5.0, 5.0, 5.0, 2.0, 5.0, 3.0, 1.0, 5.0, 2.0, 3.0, 3.0, 2.0, 3.0, 3.0, 3.0, 5.0, 1.0, 2.0, 3.0, 3.0, 2.0, 3.0, 3.0, 5.
0, 3.0, 2.0, 3.0, 2.0, 5.0, 5.0, 5.0, 3.0, 5.0, 5.0, 3.0, 2.0, 2.0, 2.0, 1.0, 5.0, 3.0, 3.0, 5.0, 3.0, 5.0, 3.0, 3.0, 2.0, 5.0, 3.0, 5.0, 3.0, 2.0, 3.0, 2.0, 2.0, 5.0,
5.0, 3.0, 2.0, 3.0, 2.0, 5.0, 5.0, 2.0, 3.0, 2.0, 2.0, 3.0, 2.0, 5.0, 2.0, 2.0, 3.0, 3.0, 3.0, 2.0, 1.0, 3.0, 3.0, 5.0, 2.0, 5.0, 3.0, 3.0, 5.0, 2.0, 5.0, 5.0, 3.0, 3.
0, 2.0, 5.0, 5.0, 2.0, 3.0, 2.0, 1.0, 5.0, 2.0, 3.0, 2.0, 5.0, 3.0, 3.0, 3.0, 3.0, 5.0, 5.0, 5.0, 5.0, 2.0, 3.0, 5.0, 3.0, 5.0, 3.0, 3.0, 5.0, 3.0, 5.0, 5.0, 3.0, 3.0,
5.0, 2.0, 3.0, 2.0, 3.0, 5.0, 2.0, 3.0, 3.0, 3.0, 3.0, 3.0, 5.0, 5.0, 3.0, 3.0, 2.0, 3.0, 3.0, 3.0, 5.0, 2.0, 5.0, 2.0, 3.0, 3.0, 5.0, 5.0, 3.0, 5.0, 3.0, 5.0, 2.0, 5.
0, 2.0, 3.0, 5.0, 3.0, 2.0, 2.0, 3.0, 3.0, 5.0, 3.0, 5.0, 2.0, 2.0, 5.0, 5.0, 5.0, 2.0, 3.0, 5.0, 2.0, 3.0, 3.0, 5.0, 3.0, 3.0, 2.0, 5.0, 2.0, 2.0, 1.0, 3.0, 5.0, 3.0,
3.0, 2.0, 3.0, 3.0, 2.0, 3.0, 3.0, 5.0, 5.0, 2.0, 2.0, 5.0, 1.0, 2.0, 5.0, 3.0, 3.0, 3.0, 2.0, 5.0, 5.0, 1.0, 2.0, 2.0, 3.0, 5.0, 5.0, 3.0, 3.0, 5.0, 3.0, 3.0, 3.0, 5.
0, 3.0, 5.0, 5.0, 1.0, 3.0, 5.0, 2.0, 2.0, 5.0, 2.0, 2.0, 3.0, 2.0, 2.0, 5.0, 5.0, 2.0, 3.0, 3.0, 3.0, 3.0, 5.0, 2.0, 5.0, 5.0, 5.0, 2.0, 3.0, 5.0, 3.0, 3.0, 2.0, 2.0,
5.0, 5.0, 3.0, 3.0, 5.0, 3.0, 3.0, 2.0, 2.0, 3.0, 5.0, 2.0, 1.0, 2.0, 5.0, 5.0, 3.0, 3.0, 5.0, 5.0, 5.0, 3.0, 2.0, 2.0, 5.0, 3.0, 5.0, 2.0, 5.0, 3.0, 5.0, 5.0, 5.0, 3.
0, 3.0, 2.0, 2.0, 3.0, 2.0, 3.0, 3.0, 2.0, 5.0, 1.0, 3.0, 5.0, 2.0, 3.0, 3.0, 2.0, 3.0, 3.0, 2.0, 5.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 2.0, 2.0, 5.0, 3.0, 3.0, 5.0, 3.0,
5.0, 5.0, 5.0, 3.0, 3.0, 5.0, 5.0, 3.0, 2.0, 3.0, 5.0, 3.0, 5.0, 2.0, 5.0, 3.0, 3.0, 1.0, 2.0, 1.0, 5.0, 5.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 2.0, 5.0, 5.0, 5.0, 5.0, 5.
0, 2.0, 3.0, 5.0, 2.0, 3.0, 5.0, 2.0, 3.0, 3.0, 3.0, 5.0, 1.0, 5.0, 5.0, 5.0, 5.0, 5.0, 5.0, 5.0, 5.0, 2.0, 2.0, 3.0, 5.0, 5.0, 5.0, 5.0, 5.0, 2.0, 5.0, 3.0, 5.0, 3.0,
5.0, 2.0, 2.0, 5.0, 3.0, 2.0, 5.0, 2.0, 3.0, 5.0, 3.0, 2.0, 3.0, 5.0, 3.0, 2.0, 5.0, 5.0, 5.0, 5.0, 2.0, 3.0, 5.0, 2.0, 5.0, 2.0, 2.0, 2.0, 1.0, 2.0, 3.0, 3.0, 5.0, 2.
0, 5.0, 5.0, 2.0, 3.0, 5.0, 5.0, 3.0, 3.0, 2.0, 2.0, 5.0, 3.0, 5.0, 5.0, 1.0, 1.0, 2.0, 5.0, 2.0, 5.0, 3.0, 2.0, 2.0, 3.0, 3.0, 1.0, 5.0, 1.0, 3.0, 5.0, 5.0, 5.0, 3.0,
3.0, 2.0, 3.0, 2.0, 3.0, 5.0, 2.0, 2.0, 5.0, 5.0, 2.0, 2.0, 3.0, 2.0, 5.0, 3.0, 3.0, 2.0, 3.0, 5.0, 2.0, 5.0, 2.0, 5.0, 3.0, 3.0, 3.0, 3.0, 2.0, 5.0, 1.0, 2.0, 5.0, 3.
0, 3.0, 3.0, 3.0, 5.0, 2.0, 3.0, 2.0, 5.0, 2.0, 5.0, 1.0, 5.0, 5.0, 2.0, 3.0, 3.0, 3.0, 5.0, 5.0, 3.0, 3.0, 5.0, 2.0, 2.0, 2.0, 3.0, 3.0, 3.0, 3.0, 5.0, 3.0, 3.0, 5.0,
3.0, 5.0, 2.0, 5.0, 2.0, 2.0, 5.0, 5.0, 3.0, 3.0, 2.0, 3.0, 5.0, 5.0, 5.0, 5.0, 3.0, 3.0, 5.0, 1.0, 5.0, 3.0, 3.0, 5.0, 2.0, 5.0, 5.0, 2.0, 2.0, 2.0, 5.0, 3.0, 3.0, 5.
0, 2.0, 2.0, 3.0, 1.0, 3.0, 2.0, 3.0, 5.0, 5.0, 3.0, 3.0, 5.0, 5.0, 5.0, 3.0, 5.0, 3.0, 5.0, 3.0, 3.0, 3.0, 3.0, 5.0, 2.0, 2.0, 3.0, 5.0, 3.0, 3.0, 5.0, 2.0, 3.0, 3.0,
3.0, 3.0, 5.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 5.0, 3.0, 5.0, 2.0, 2.0, 3.0, 3.0, 5.0, 5.0, 5.0, 3.0, 5.0, 2.0, 3.0, 5.0, 5.0, 3.0, 2.0, 5.0, 5.0, 2.0, 5.
0, 5.0, 3.0, 5.0, 5.0, 3.0, 3.0, 5.0, 3.0, 5.0, 3.0, 3.0, 3.0, 5.0, 3.0, 3.0, 5.0, 2.0, 3.0, 2.0, 3.0, 3.0, 5.0, 3.0, 3.0, 3.0, 2.0, 2.0, 3.0, 2.0, 2.0, 3.0, 2.0, 3.0,
2.0, 1.0, 3.0, 5.0, 3.0, 5.0, 5.0, 5.0, 2.0, 3.0, 2.0, 5.0, 3.0, 5.0, 3.0, 5.0, 5.0, 1.0, 5.0, 3.0, 5.0, 3.0])
Accuracy: 20.101010101%
>>> |
```



```
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:43:06) [MSC v.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
Enter the user ID:54
Connected_users: ['57', '58', '73', '2016', '2017', '2018', '2019', '2020', '2021', '2022', '2023', '2024', '2025', '2026', '2027', '2028', '2029', '2030', '2031', '20
32', '2033', '2034', '2035', '2036', '2037', '2038', '2039', '2040', '2041', '2042', '2043', '2044', '2045', '2046', '2047', '2048', '2049', '2050', '2051', '2052', '2
053', '2055', '2060', '2061', '2063', '2519', '3772', '3861']
Recommended subscriptions: ['6911', '6673', '6851', '6588', '6061', '5320', '6345', '6918', '6590', '6620', '5934', '6055', '5332', '5858', '6132', '5223', '5813', '62
24', '5461', '6259', '5669', '5586', '5117', '5756', '6520', '5731', '6905', '5738', '5719', '5879', '5570', '6579', '6496', '6413', '6100', '6559', '6411', '5241', '6
500', '6478', '5048', '5562', '6643', '6624', '6448', '6576']
>>>
```

6. Applications

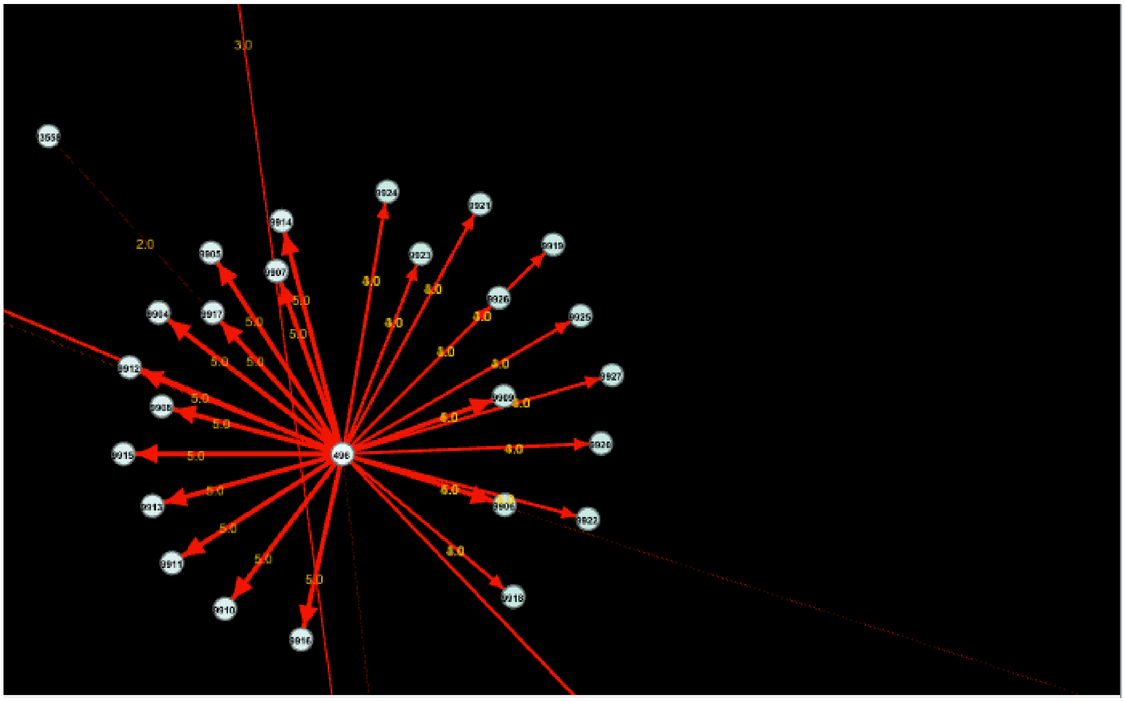
From the datasets obtained there are two types of data i.e. video statistics and user statistics. Video statistics obtained have the attributes uploader, video length, video size and date added.

The user statistics have the following attributes like user id, relation/connection with other users and location.

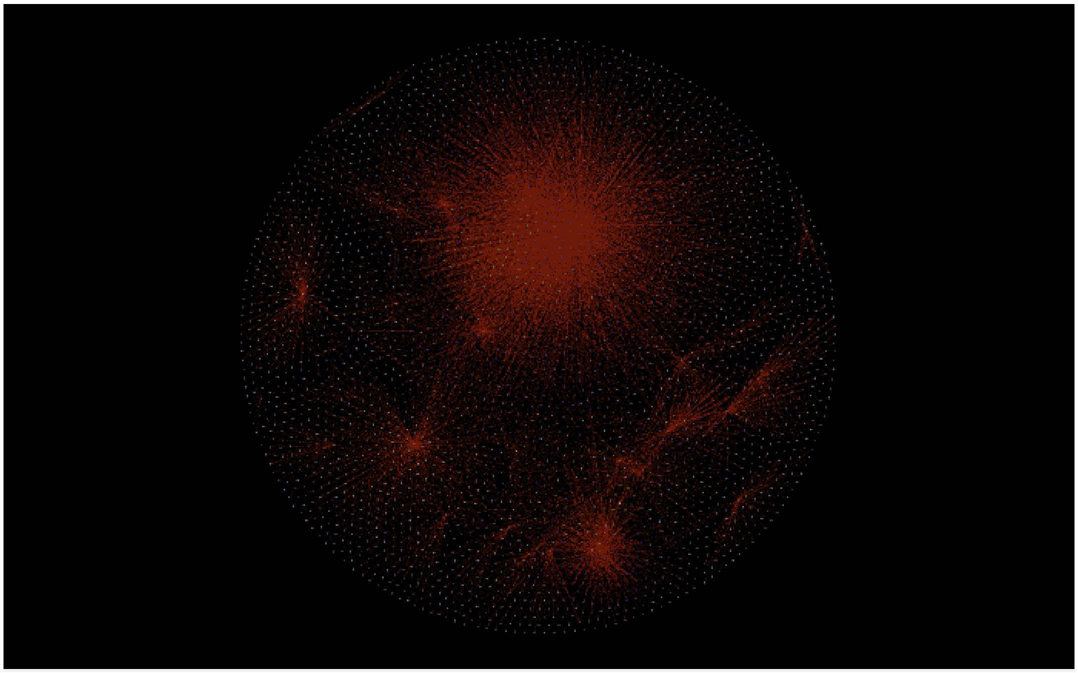
These data sets will be used to form graphs which have users and videos as nodes and use connections between users and the videos uploaded by them as edges. This way we can use directed graphs to visualize the data obtained for easier analysis.

Screenshots:

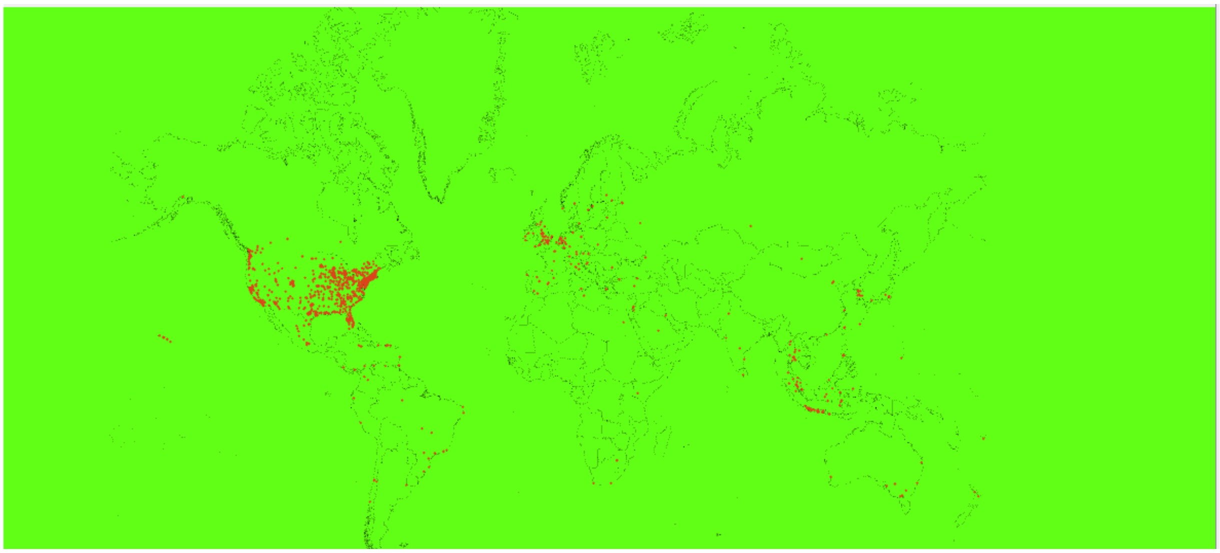
Rating by users



Connection between users or shared friends



User locations



7. Tools/programming language

To create the graphs and regression models, tools and programming languages like python and R will be used. To form the graphs various libraries of Python such as Matplotlib will be used. For creating the regression models and studying them R can be used.

Also, additional help from Gephi tool might be needed for some datasets.

8. Conclusion

After various test case runs we can safely conclude that the developed algorithm is a good method to recommend channels to users on YouTube based on their social connections and subscriptions. It can also be assumed that the prediction algorithm helps predict channel ratings by users. Graphs developed using the Gephi tool help us visualize the social network of the website and make statistical analysis user which is in turn used for prediction of ratings by users and other channels.

9.References

[1] "YouTube serves up 100 million videos a day online," http://www.usatoday.com/tech/news/2006-07-16-youtube-views x.htm.

[2] "Google to buy YouTube for $1.65 billion," http://money.cnn.com/2006/10/09/technology/googleyoutube deal/ index.htm.

[3] "YouTube video-sharing site is changing popular culture," http://www.kcrw.com/news/programs/ww/ww061122youtube video-sharin.

[4] "Alexa," http://www.alexa.com. [5] "YouTube: Video Format (Wikipedia)," http://en.wikipedia.org/wiki/Youtube#Video format.

[6] "API Documentation (YouTube)," http://youtube.com/dev docs.

[7] W. Tang, Y. Fu, L. Cherkasova, and A. Vahdat, "Long-term Streaming Media Server Workload Analysis and Modeling," HP Labs, Tech. Rep., 2003.

[8] "YouTube Blog," http://youtube.com/blog.

[9] S. Acharya, B. Smith, and P. Parnes, "Characterizing User Access To

Videos On The World Wide Web," in Proc. of ACM/SPIE Multimedia Computing and Networking, 2000.