Technical Appendix

Catch the Pink Flamingo Analysis

Produced by: Jean André Marcel Calestani

Acquiring, Exploring and Preparing the Data

Data Exploration

Data Set Overview

The table below lists each of the files available for analysis with a short description of what is found in each one.

File Name	Description	Fields
ad-clicks.csv	A line is added to this file when a player clicks on an advertisement in the Flamingo app.	timestamp: when the click occurred. txID: a unique id (within ad-clicks.log) for the click userSessionid: the id of the user session for the user who made the click teamid: the current team id of the user who made the click userid: the user id of the user who made the click adID: the id of the ad clicked on adCategory: the category/type of ad clicked on
buy-clicks.csv	A line is added to this file when a player makes an in-app purchase in the Flamingo app.	timestamp: when the purchase was made. txID: a unique id (within buy-clicks.log) for the purchase userSessionid: the id of the user session for the user who made the purchase team: the current team id of the user who made the purchase

		userid: the user id of the user who made the purchase buyID: the id of the item purchased price: the price of the item purchased
users.csv	This file contains a line for each user playing the game.	timestamp: when user first played the game. id: the user id assigned to the user. nick: the nickname chosen by the user. twitter: the twitter handle of the user. dob: the date of birth of the user. country: the two-letter country code where the user lives.
team.csv	This file contains a line for each team terminated in the game.	teamid: the id of the team name: the name of the team teamCreationTime: the timestamp when the team was created teamEndTime: the timestamp when the last member left the team strength: a measure of team strength, roughly corresponding to the success of a team currentLevel: the current level of the team
team-assignments.csv	A line is added to this file each time a user joins a team. A user can be in at most a single team at a time.	time: when the user joined the team. team: the id of the team userid: the id of the user

		assignmentid: a unique id for this assignment
level-events.csv	A line is added to this file each time a team starts or finishes a level in the game	time: when the event occurred. eventid: a unique id for the event teamid: the id of the team level: the level started or completed eventType: the type of event, either start or end
user-session.csv	Each line in this file describes a user session, which denotes when a user starts and stops playing the game. Additionally, when a team goes to the next level in the game, the session is ended for each user in the team and a new one started.	timeStamp: a timestamp denoting when the event occurred. userSessionId: a unique id for the session. userId: the current user's ID. teamId: the current user's team. assignmentId: the team assignment id for the user to the team. sessionType: whether the event is the start or end of a session. teamLevel: the level of the team during this session. platformType: the type of platform of the user during this session.
game-clicks.csv	A line is added to this file each time a user performs a click in the game.	time: when the click occurred. clickid: a unique id for the click. userid: the id of the user performing the click. usersessionid: the id of the session of the user when the click is performed.

		isHit: denotes if the click was on a flamingo (value is 1) or missed the flamingo (value is 0) teamId: the id of the team of the user teamLevel: the current level of the team of the user

Aggregation

Amount spent buying items	21407
# Unique items available to be purchased	6

Analysis on the # of unique items available:
The number of 6 items to be purchased is very limited. Users, especially gamers here, like to collect items. I would suggest to augment fiercely this number.

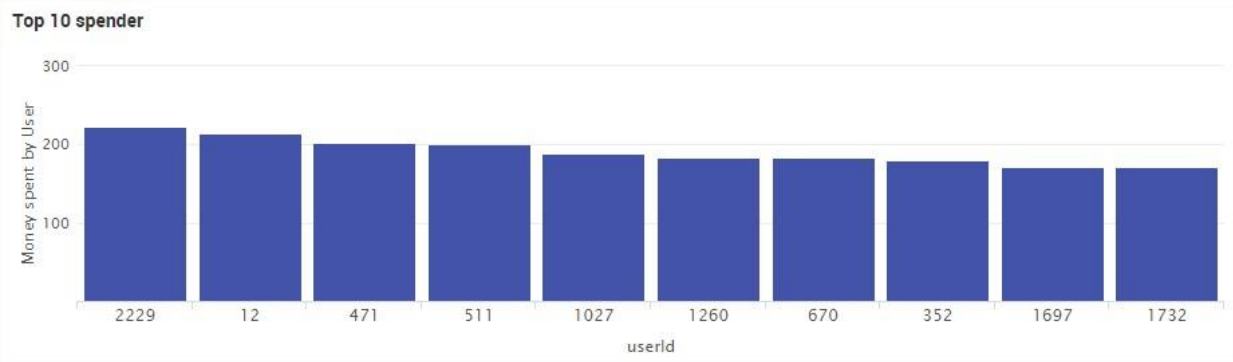
A histogram showing how many times each item is purchased:

A histogram showing how much money was made from each item:

Analysis on the two above histograms: the repartition of the items bought and the amount spent by item are quite unequal. I would suggest to advertise more on the items less bought and average or quasi-equalize the price of the items.

Filtering

A histogram showing total amount of money spent by the top ten users (ranked by how much money they spent).



The following table shows the user id, platform, and hit-ratio percentage for the top three buying users:

Rank	User Id	Platform	Hit-Ratio (%)
1	2229	iphone	0.115970
2	12	iphone	0.130682
3	471	iphone	0.145038

Analysis of the filtering tests:

The top 10 users by money spent or spenders have an average of 200 spent. The 3 top spenders seem to be iPhone owner and hit-ratio is going from 11% to 14%. I would suggest to expand the last analysis to more users to get a better sense of statistics. If we want, anyway, to conclude something of that last table: I would suggest

1/ to make more advertisement on Windows and Android users

and

2/ to lower the difficulty of the game as 11% or 16% of success hit shows a great difficulty to aim at the targets/flamingos. An easier game is played by more people, is more accessible.

Data Classification Analysis

Data Preparation

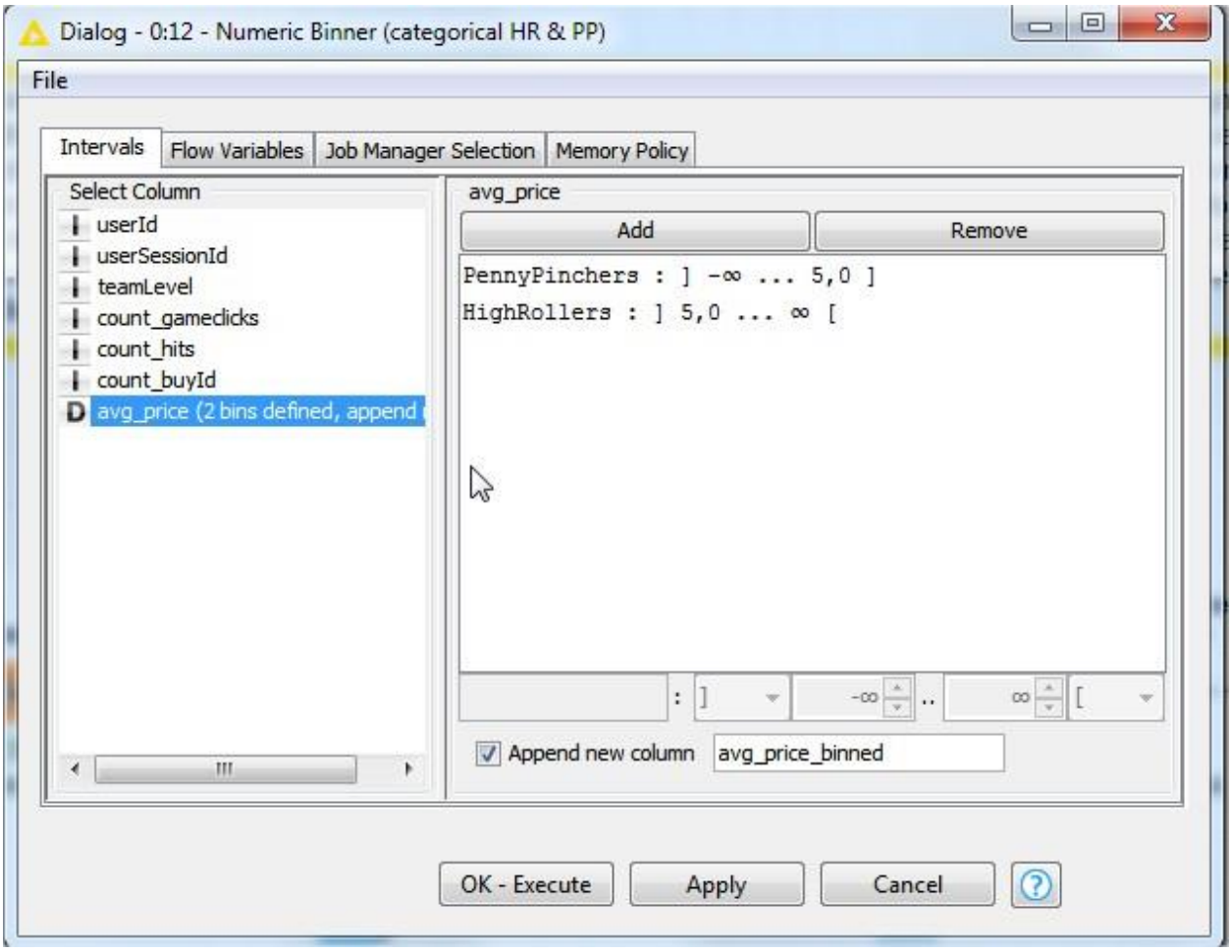
Analysis of combined_data.csv

Sample Selection

Item	Amount
# of Samples	4619
# of Samples with Purchases	1411

Attribute Creation

A new categorical attribute was created to enable analysis of players as broken into 2 categories (HighRollers and PennyPinchers). A screenshot of the attribute follows:



The goal of this new attribute will separate the gamers into two categories : the first ones, PeenyPinchers, who usually spent 5$ or less and the second ones, HighRollers, who spent usually at least 5$.

The creation of this new categorical attribute was necessary because predicting which users will spent more or less money in the in-app purchases is a valuable information for Eglence Inc.

Attribute Selection

The following attributes were filtered from the dataset for the following reasons:

Attribute	**Rationale for Filtering**

Avg-price	The attribute doesn't depend directly of the user
userId	Information originating from our server
Usersessionid	Information originating from our server

Data Partitioning and Modeling

The data was partitioned into train and test datasets.

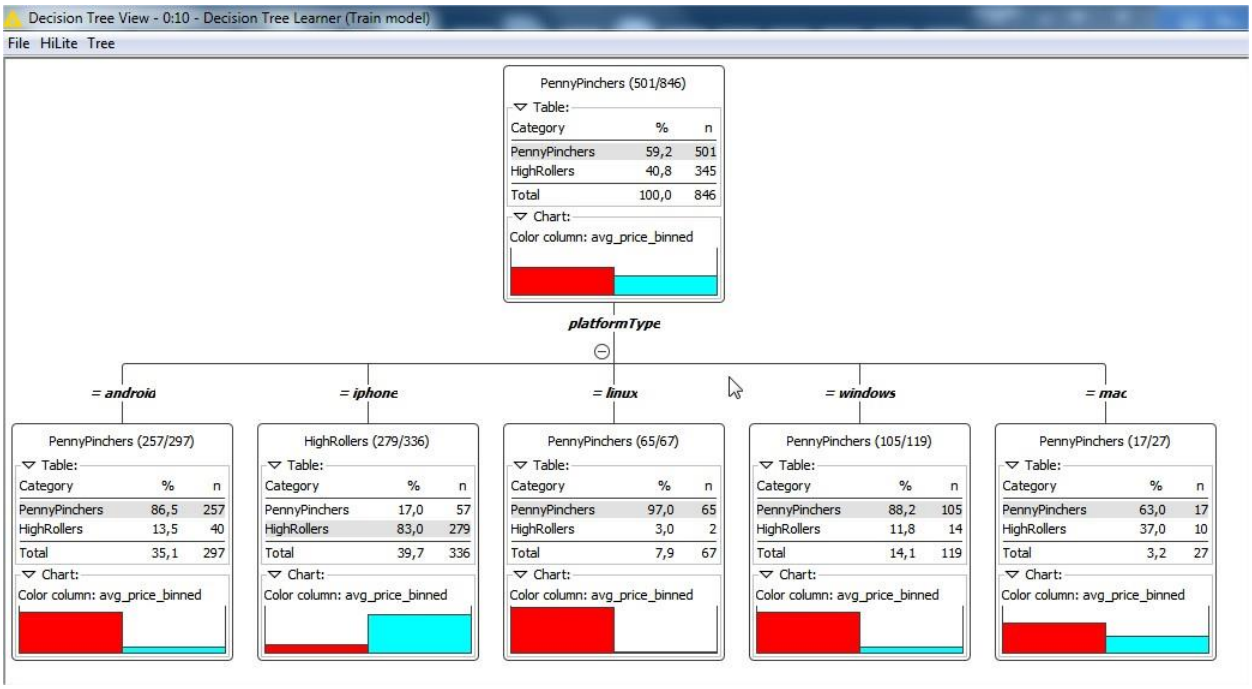
The trained data set was used to create the decision tree model.

The trained model was then applied to the test dataset.

This is important because want to test our model on data that was not used to train it.

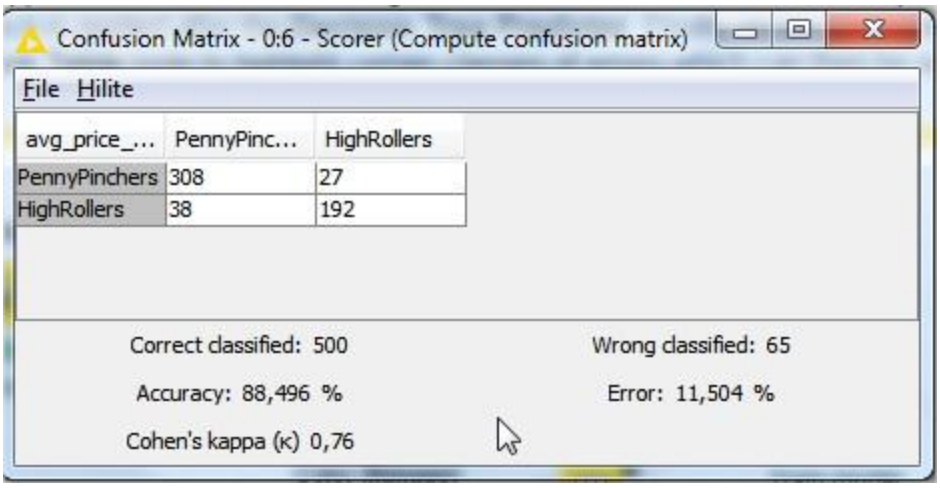
When partitioning the data using sampling, it is important to set the random seed because it ensures that we get reproducible results.

A screenshot of the resulting decision tree can be seen below:



Evaluation

A screenshot of the confusion matrix can be seen below:



As seen in the screenshot above, the overall accuracy of the model is 88.496%.

Meaning for each of the values of the confusion matrix:

-Upper left corner: represent the True Positive: meaning the predicted value is equal to the tested value and both values are Positive. Here it represents the true 308 Pennypinchers.

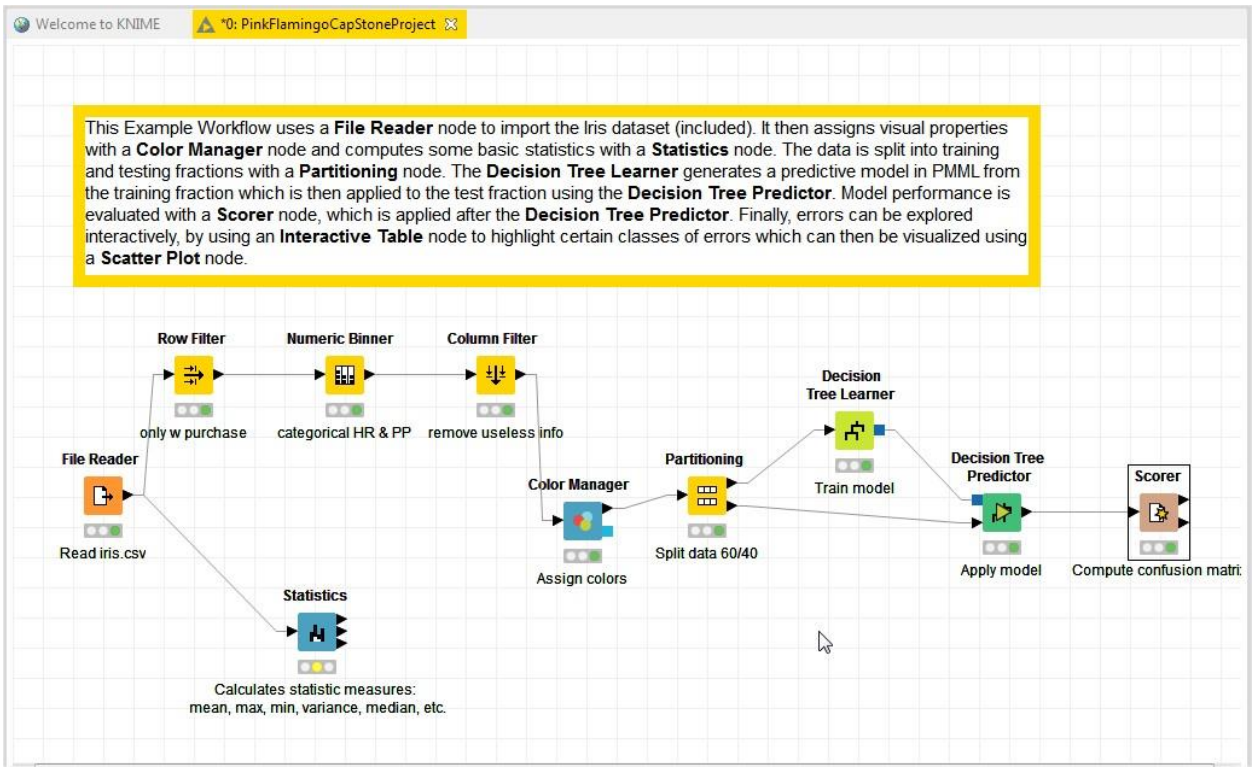
-Lower left corner: represent the False Negative: meaning the predicted value is different from the tested value and the predicted value is Negative and the tested value is Positive. Here it represents the 38 false HighRollers.

-Upper right corner: represent the False Positive: meaning the predicted value is different from the tested value and the predicted value is Positive and the tested value is Negative. Here it represents the 27 false Pennypinchers.

-Lower right corner: represent the True Negative: meaning the predicted value is equal to the tested value and both values are Negative. Here it represents the true 192 HighRollers.

Analysis Conclusions

The final KNIME workflow is shown below:



What makes a HighRoller vs. a PennyPincher?

The decision Tree shows clearly that the vast majority of HighRollers belongs to iPhone owners. Around 83% of them are HighRollers. We can note and add that the Mac's owners are representative too of the category HighRollers as 37% of them belong to this category.

Specific Recommendations to Increase Revenue
1. Too increase the revenue we recommend first to inforce the advertisement on the Apple Store.
2. Secondly we recommend to augment the number of possibilities of in-app purchases on the others platforms: Linux, android and windows platforms.

Clustering Analysis

Attribute Selection

Attribute	Rationale for Selection
Sum of Ad Clicks by user	Here we compare user by the sum of their click on the ads inside the game. Is there any group/s of user who click more than another on the add?
Sum of Money Spent by User and Click Accuracy	Here we compare the over whole money spent by user and his ability in the game. Are the best players the big spenders?
Sum of Money Spent by the frequency of user's session	Here we compare the sum of user session by user to the money spent by user. Are the frequent users the big spenders?

Training Data Set Creation for the Sum of Ad Clicks

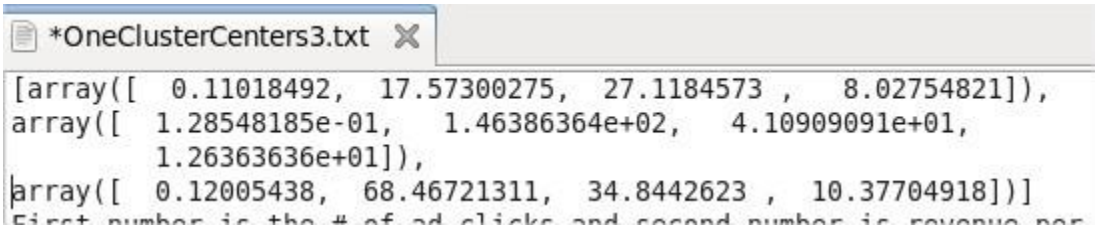
The training data set used for this analysis is shown below (first 5 lines):

```
In [27]: trainingDF4.shape
Out[27]: (529, 4)

In [28]: trainingDF4.head(5)
Out[28]:
      isHit  price  adCount  sessionCount
0  0.134078   21.0       44            14
1  0.100000   53.0       10             4
2  0.122047   80.0       37            12
3  0.109430   11.0       19            10
4  0.130682  215.0       46            14

In [29]:
```

Dimensions of the training data set (rows x columns): 529 x 4

# of clusters created: 3

Cluster Centers



Cluster #	Cluster Center
1	[0.11018492, 17.57300275, 27.1184573, 8.02754821]
2	[0.128548185, 146.386364, 41.0909091, 12.6363636]
3	[0.12005438, 68.46721311, 34.8442623, 10.37704918])]

These clusters can be differentiated from each other as follows:

Cluster 1 is different from the others in that it has the lowest revenue per user, the number of ad-clicks per user, the lowest number of session per user. The first term of my centers is the accuracy which is quite always the same here (and also on others calculations made with 4 and 5 centers) so I won't use it in my analysis as it seems irrelevant. This center shows us the users who spend the less and use the less the game. We have to keep in mind that these users might be the newest ones.

Cluster 2 is different from the others in that it shows us the best users for Eglence Inc. in term of revenue as these ones play the most and spend the most.

Cluster 3 is different from the others in that it shows us an intermediate group of users: they spend half of the previous group but 4 times more than the first group. They click quite a lot on the ads in comparison on what they spend and played an average number of session.

Recommended Actions

Action Recommended	**Rationale for the action**
Make a promotion to new users	Going from 8 sessions to 10: the revenue is multiply by 4. Eglence should highly encourage the users who have around 8 sessions to continue. Big bonuses if the player continue until his 10^{th} session. The same would be efficient from the players who have 10 sessions and the ones who have almost 13 as the revenue is, this time, multiply by 2.
Make attractive ads to average users	As the number of ad-clicks, between average spenders and big spenders, increased by 20% the revenue is multiply by 200%. To better the quality, survey and watch out the ads displayed for this group seems important.

Graph Analytics Analysis

Graph Analytics

Modeling Chat Data using a Graph Data Model

The graph can be described as follow: a user (User) can create a chat (CreateChat), this chat (ChatItem) is a part of (PartOf) a team chat session (TeamChatSession) owned by (OwnedBy) a team (Team).

Also a user (User) can create (CreatesSession), join (Joins) or leave (Leaves) a team chat session (TeamChatSession).

Inside a chat (ChatItem) a user (User) can be mentioned (Mentioned). And finally to answer (ResponseTo) to a chat (ChatItem) is also a chat (ChatItem).

Convention:
User : all names painted in yellow are Nodes of the graph.
CreatesSession : all names painted in green are Edges of the graph.

Creation of the Graph Database for Chats

Describe the steps you took for creating the graph database. As part of these steps

i) The schema of the 6 CSV files:

File: chat_create_team_chat.csv
userid : Id of the user creating a new ChatSession node
teamid : Id of the team which the user belongs to
teamchatsessionid : id of the newly chat session created
timestamp : timestamp of the creation of this new node.

File: chat_item_team_chat.csv
userid : Id of the user creating a new ChatItem node
teamchatsessionid : Id of the team chat session rattached to this ChatItem Node
chatItemid : id of the newly chat item created
timestamp : timestamp of the creation of this new node. Will serve to generate 2 edges: CreateChat and PartOf respectively coming from a User node and TeamSessionChat node.

File: chat_join_team_chat.csv
userid : : Id of the user joinng a new ChatSession
TeamChatSessionID : Chatsession id that is joined by the user
timestamp : timestamp of the creation of this new edge.

File: chat_leave_team_chat.csv
userid : Id of the user leaving a new ChatSession
TeamChatSessionID : Chatsession id that is left by the user

timestamp : timestamp of the creation of this new edge.

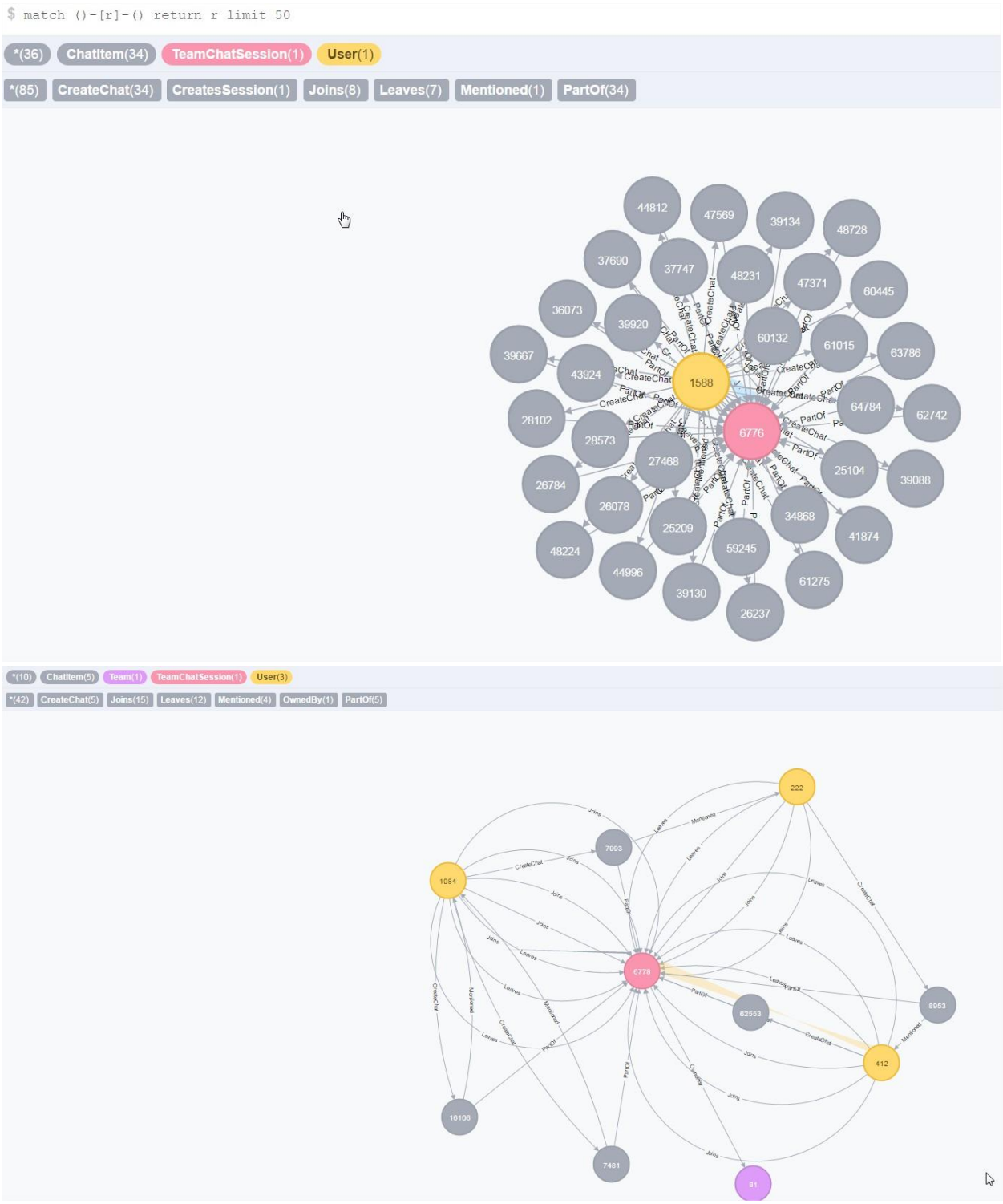
File: chat_mention_team_chat.csv
ChatItem : id from the chat item that mentioned an user
userid : user id mentioned by the Chat Item
timestamp : used to make the edge between the chatitem and the user node.

File: chat_respond_team_chat.csv
userid1 : user that create a new chatitem to respond to another second user
userid2 : that second user
timestamp : used to make the edge between the chatitems.

ii) The loading process consists of three main steps:
1/Naming the source of the data through a CSV file
2/Adding the Nodes using the MERGE command
3/Adding the Edges using also the MERGE command.
LOAD Command for the chat_join_team_chat.csv : :

```
LOAD CSV FROM
"file:///c:/Users/jean/Desktop/Big%20Data%20Cours/Course%206%20Capstone%20Project/
w4/chat%20data/chat_join_team_chat.csv" AS row
MERGE (u:User {id: toInt(row[0])})
MERGE (c:TeamChatSession {id: toInt(row[1])})
MERGE (u)-[:Joins{timeStamp: row[2]}]->(c)
```

iii) Screenshots of some part of the graph generated:



Finding the longest conversation chain and its participants

Report the results including the length of the conversation (path length) and how many unique users were part of the conversation chain. Describe your steps. Write the query that produces the correct answer.

The longest conversation chain based on the “ResponseTo” edge is composed of 10 nodes and 9 edges. The query is composed by a MATCH including all nodes connected by the ResponseTo edge then it’s ordered by the length of all paths founds in descending order.

The participants of the longest conversation chain can be found by using the first query and adding the command “With P” and an another MATCH to count the distinct users who create a chat using only the edges [CreateChat]. They are 5 users. Their ID are the following: 1192, 853, 1514, 1978, 1153.

The query to the longest conversation chain:

```
match p=()-[:ResponseTo*]-() return p order by length(p) desc limit 1.
```

And its participants:

```
match p=()-[:ResponseTo*]-() where length(p)=9

with p

match (u:User)-[c:CreateChat*]-(i:ChatItem)

where i.id in EXTRACT(n IN NODES(p)| n.id)

return u as Users, p, c

limit 25.
```

Analyzing the relationship between top 10 chattiest users and top 10 chattiest teams

The first query is simple: we match any user who have any number of edges called CreateChat and then ordered them by the numbers of these edges in the descending order to get the chattiest first in the list of results.

The query to get the 10 chattiest users:

```
match p2=(u)-[r:CreateChat]-(i2)

return u as UserS, count(distinct r) as rel

order by rel desc limit 10
```

Chattiest Users

Users	Number of Chats

394	115
2067	111
1087	109

The query to get the 10 chattiest teams :

match p=(i)-[r:PartOf]-(:TeamChatSession)-[r2:OwnedBy]-(t:Team)

return t as Team, count(distinct i) as chatitems

order by chatitems desc limit 10

Chattiest Teams

Teams	Number of Chats
82	1324
185	1036
112	957

Is there any chattiest user in the chattiest team?

Let's execute the following query:

Match (u:User)-[:Joins]-(:TeamChatSession)-[o:OwnedBy]-(t:Team)

where t.id in ([82,185,112,18,194,129,52,136,146,81])

AND u.id in ([394,2067,1087,209,554,1627,516,999,461,668])

return t as Team, u as Users, o limit 1000

Finally, just one user, user.id = 999 belongs to the team, team.id = 52.

How Active Are Groups of Users?

Describe your steps for performing this analysis. Be as clear, concise, and as brief as possible. Finally, report the top 3 most active users in the table below.
First we will find the neighbors of the 10th chattiest users thanks to this query:

Ex. Done with user 209:
match (u)-[r:InteractsWith]-(u2)
where u.id=209
return u, u2,r

To get only one relationship InteractsWith between two nodes, the following query is passed :
match (s)-[r:InteractsWith]->(e)

```
with s,e,type(r) as typ, tail(collect(r)) as coll
foreach(x in coll | delete x)
```

From that we get the number of relationship between the neighbors.
Now as we have both numbers of neighbors and relationships between them we can calculate we cluster coefficient of each.

Most Active Users (based on Cluster Coefficients)

User ID	**Coefficient**
209 (8 users in the cluster and 54 relationships)	0,9642857142857143
554 (8 users in the cluster and 48 relationships)	0,8571428571428571
1087 (7 users in the cluster and 35 relationships)	0.8333333333333333