Technical Appendix
Catch the Pink Flamingo Analysis

Produced by: Ziqing Wang

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	**ERD table: AdClicks** 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	**ERD table: InAppPurchases** 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	**ERD table: User** This file contains a line for each user playing the game.	timestamp: when user first played the game. userId: 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	**ERD table: Team** 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

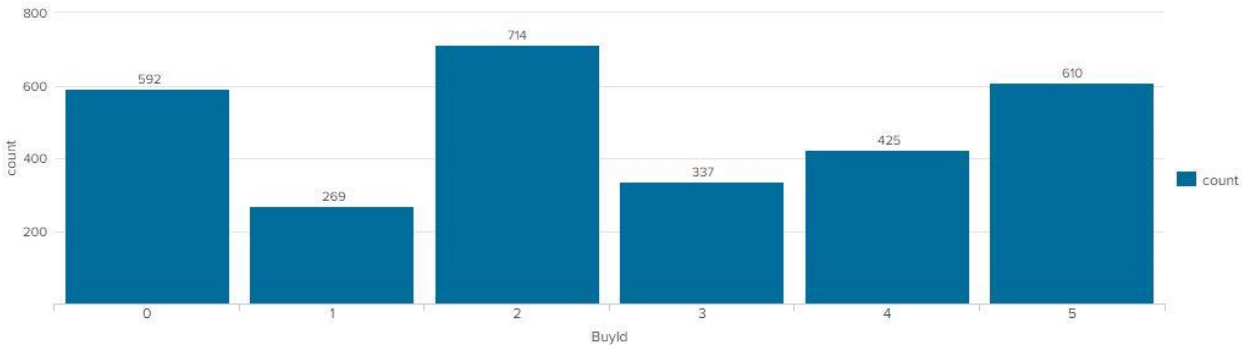
		currentLevel: the current level of the team
team-assignments.csv	**ERD table: TeamAssignment** A line is added to this file each time a user joins a team. A user can be in at most a single team at a time.	timestamp: when the user joined the team. team: the id of the team userId: the id of the user assignmentId: a unique id for this assignment
level-events.csv	**ERD table: LevelEvent** A line is added to this file each time a team starts or finishes a level in the game	timestamp: when the event occurred. eventId: a unique id for the event teamId: the id of the team teamLevel: the level started or completed eventType: the type of event, either start or end
user-session.csv	**ERD table: User_Sessions** 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	**ERD table: GameClicks** A line is added to this file each time a user performs a click in the game.	timestamp: 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

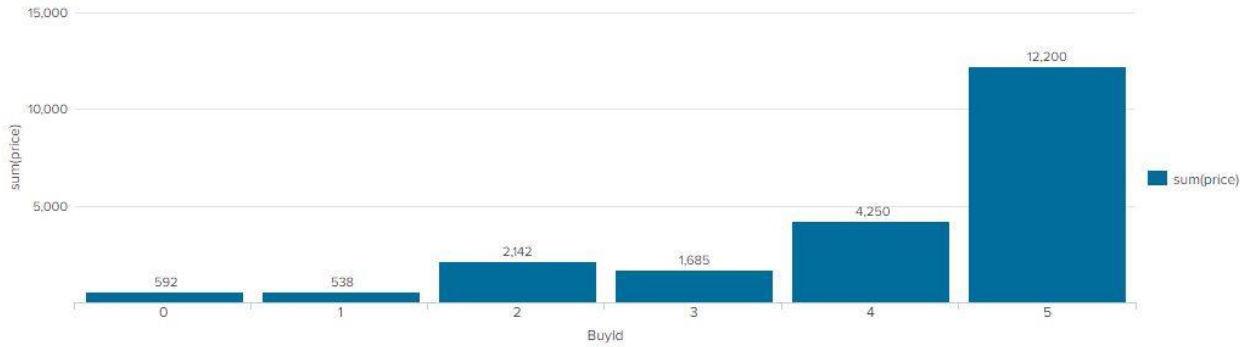
Aggregation

Amount spent buying items	21407
Number of unique items available to be purchased	6

A histogram showing how many times each item is purchased:

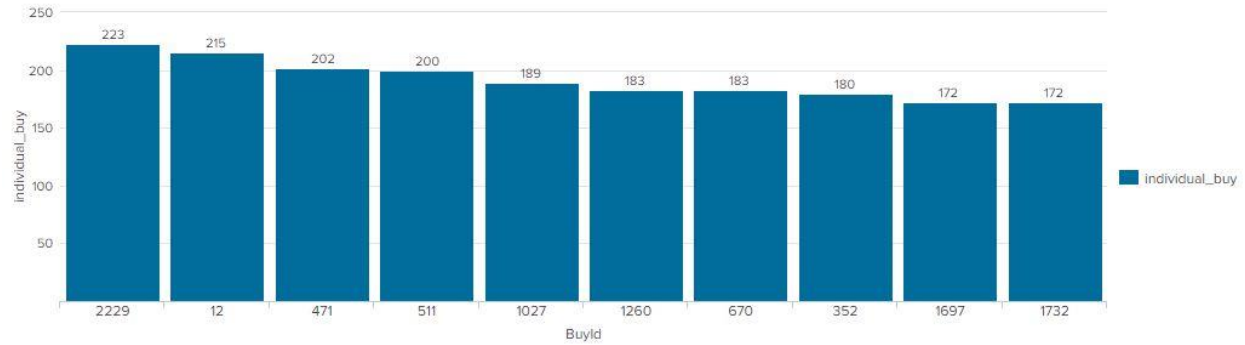


A histogram showing how much money was made from each item:



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	11.60
2	12	iphone	13.07
3	471	iphone	14.50

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:

Row ID	teamLevel	platfor...	count_...	count_...	count_buyId	spending_pattern
Row4	1	android	39	0	1	PennyPinchers
Row11	1	iphone	129	9	1	HighRollers
Row13	1	android	102	14	1	PennyPinchers
Row17	1	android	39	4	1	PennyPinchers
Row18	1	android	90	10	1	PennyPinchers
Row31	1	iphone	51	8	1	HighRollers
Row49	1	android	51	6	2	PennyPinchers
Row50	1	android	47	5	2	PennyPinchers
Row58	1	android	46	7	1	PennyPinchers
Row61	1	iphone	41	6	1	HighRollers
Row68	1	android	47	7	1	PennyPinchers
Row72	1	iphone	76	7	1	HighRollers
Row73	1	android	52	2	1	PennyPinchers
Row101	1	android	62	9	1	PennyPinchers
Row122	1	iphone	177	25	2	HighRollers
Row127	1	iphone	54	5	1	HighRollers
Row129	1	android	27	4	2	PennyPinchers
Row131	1	iphone	37	2	1	HighRollers

I created a new categorical attribute named “spending_pattern” using Numeric Binner node based on the value of avg_price. If the value of avg_price is less than or equal to 5, the categorical value would be PennyPinchers, else it would be HighRollers.

The creation of this new categorical attribute was necessary because different nominal values are required for the following classification by decision tree.

Attribute Selection

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

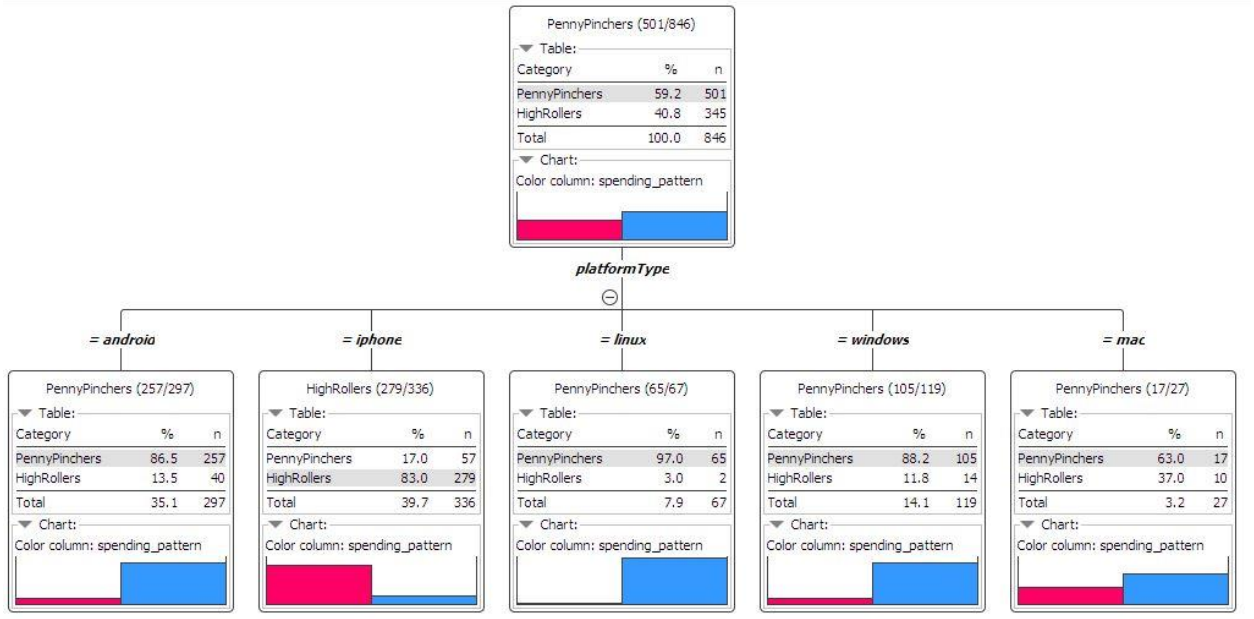
Attribute	Rationale for Filtering
userId	It is not related to the clustering
userSessionId	It is not related to the clustering
avg_price	It is not needed anymore since we've already made the new categorical attribute based on this value
<Optional Fill in>	<Optional Fill in 1-3 sentences>

Data Partitioning and Modeling

The data was partitioned into train and test datasets.
The **training** data set was used to create the decision tree model.
The trained model was then applied to the **testing** dataset.
This is important because the model has to be evaluated by unseen data.

When partitioning the data using sampling, it is important to set the random seed because the random seed can ensure the partitioning results would be the same among different students.

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



Evaluation

A screenshot of the confusion matrix can be seen below:

spending_pattern \ Prediction (spending_pattern)	PennyPinchers	HighRollers
PennyPinchers	308	27
HighRollers	38	192

Correct classified: 500 Wrong classified: 65

Accuracy: 88.496 % Error: 11.504 %

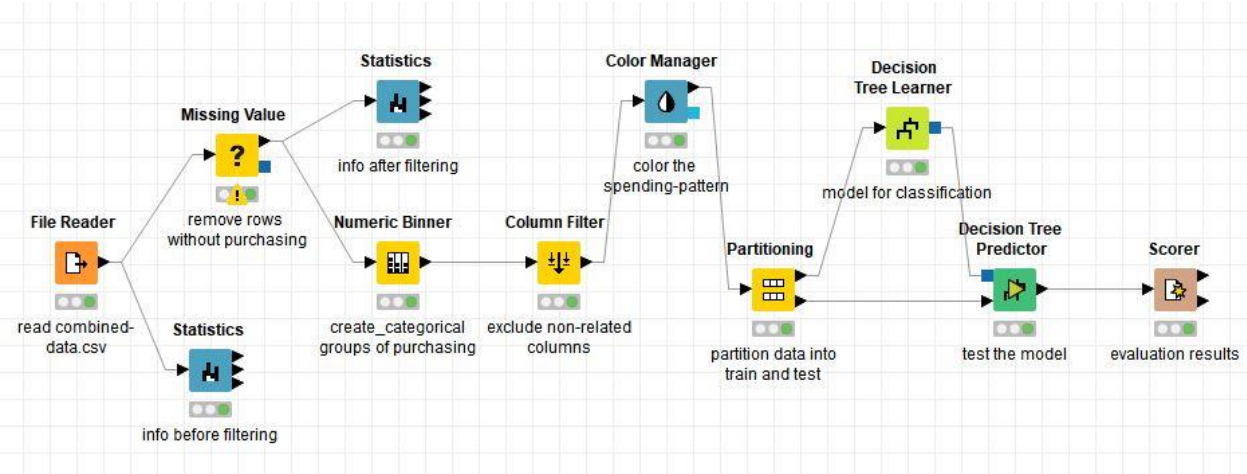
Cohen's kappa (κ) 0.76

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

308 samples with PennyPinchers were predicted as PennyPinchers correctly.
27 samples with PennyPinchers were predicted as HighRollers incorrectly.
38 samples with HighRollers were predicted as PennyPinchers incorrectly.
192 samples with HighRollers were predicted as HighRollers correctly.

Analysis Conclusions

The final KNIME workflow is shown below:



What makes a HighRoller vs. a PennyPincher?
Mobile users with iPhone contribute the most of the HighRollers. In the contrast, if users are using other platforms as android, linux, windows or mac, they are more likely to be a PennyPincher.

Specific Recommendations to Increase Revenue
1. Invest more in the game development on mobile end including both android and iPhone.
2. Try to attract more mobile users of iPhone.

Clustering Analysis

Attribute Selection

Attribute	Rationale for Selection
total number of ad-clicks per user	This attribute indicates the general interest of user for in-game advertisement, which is useful for identifying potential consumers
sum of money spent on items purchased in-game by each user	This attribute indicates the in-game purchasing tendency for different user, which is a good marker for in-game shopper
Total game clicks by each user	This attribute indicates how much the user has engaged in the game, which is useful for categorizing the users into uber-player or casual player

Training Data Set Creation

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

```
+--------------+-----------------+----------------+
|total_ad_click|  sum_money_spent|total_game_click|
+--------------+-----------------+----------------+
|            29|             10.0|            1121|
|            10|             13.0|             172|
|            39|             16.0|             670|
|            16|             55.0|             773|
|            29|39.34990791896869|              71|
+--------------+-----------------+----------------+
only showing top 5 rows
```

Dimensions of the training data set (rows x columns) : 1193 x 3

# of clusters created: 7

Cluster Centers

Cluster #	Center
1	[0.43709526, 0.01546897, 3.34640766]
2	[0.08223984, -0.01706471, -0.43020903]
3	[1.23090124, 3.24303115, 0.23318008]
4	[-0.07339694, -0.20438671, 0.95820797]
5	[1.59159633, -0.33605292, 0.13606247]

6	[-1.40789793, -1.06890625, -0.55179499]
7	[-1.68651176, 0.19335887, -0.67628608]

These clusters can be differentiated from each other as follows:

Cluster 1 is different from the others in that…

- Users are core players who click game much more frequently than the others.

Cluster 2 is different from the others in that…

- Users show average interest for ads and spend average amount of money for in-game purchasing.
- Users are one group of least active players.

Cluster 3 is different from the others in that…

- Users spend the most for in-game purchasing and click ads very frequently.

Cluster 4 is different from the others in that…

- Users play game a lot but show average interest in ads and in-game purchasing.

Cluster 5 is different from the others in that…

- Users are interested in ads much more than the other players.

Cluster 6 is different from the others in that…

- Users are not interested in everything.

Cluster 7 is different from the others in that…

- Users spend average amount of money for in-game purchasing, but are not interested at all in ads and game play.

Recommended Actions

Action Recommended	**Rationale for the action**
Improve mission rewards, down-regulate mission difficulty to attract more casual players	Most of the users are in cluster 2 which indicates a casual player base. Thus the company should take the casual players as the major target to develop the popularity.
Improve the quality of in game ads to attract more clicks	Users in cluster 3 click ads very frequently and spend most for in-game shopping. Better ads can arouse more of their interests.

Graph Analytics Analysis

Modeling Chat Data using a Graph Data Model

The graph data model here is to display the relationships between different users or different teams through their conversations (chat data) from several different sources.

Creation of the Graph Database for Chats

i. The graph is built from these 6 CSV files:

- Chat_create_team_chat.csv:
  1st column: User id,
  2nd column: Team id,
  3rd column: Team chat session id,
  4th column: Timestamp of creating team chat session.

- Chat_join_team_chat.csv:
  1st column: User id,
  2nd column: Team chat session id,
  3rd column: Timestamp of joining team chat session.

- Chat_leave_team_chat.csv:
  1st column: User id,
  2nd column: Team chat session id,
  3rd column: Timestamp of leaving team chat session.

- Chat_item_team_chat.csv:
  1st column: User id,
  2nd column: Team chat session id,
  3rd column: Chat item id,
  4th column: Timestamp of creating chat

- Chat_mention_team_chat.csv:
  1st column: Chat item id,
  2nd column: User id,
  3rd column: Timestamp of the chat item mentioning the user

- Chat_respond_team_chat.csv:
  1st column: First chat item id,
  2nd column: Second chat item id,
  3rd column: Timestamp of the first chat item responding to the second chat item

ii. LOAD command example:

```
LOAD CSV FROM "file:/path/to/chat_create_team_chat.csv" AS row
MERGE (u:User {id: toInteger(row[0])}) MERGE (t:Team {id: toInteger(row[1])})
MERGE (c:TeamChatSession {id: toInteger(row[2])})
```

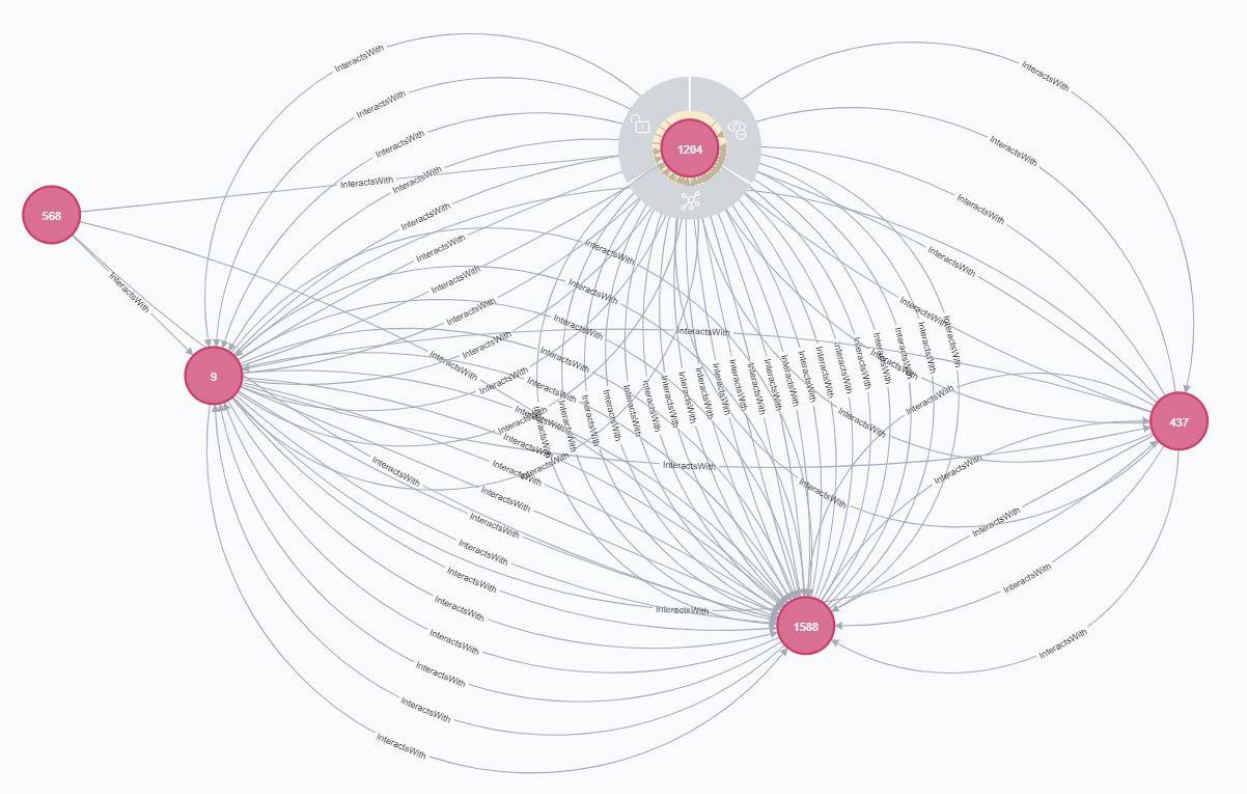
MERGE (u)-[:CreatesSession{timeStamp: row[3]}]->(c)
MERGE (c)-[:OwnedBy{timeStamp: row[3]}]->(t)

The first line gives the path of the file. This command reads the chat_create_team_chat.csv file one row at a time and creates User nodes. The 0th column value is converted to an integer and is used to populate the id attribute. Similarly the other nodes are created.

Line 4, MERGE (u)-[:CreatesSession{timeStamp: row[3]}]->(c) creates an edge labeled "CreatesSession" between the user node u and the TeamChatSession node c. This edge has a property called timeStamp. This property is filled by the content of column 3 of the same row.

Similarly, the last line creates an edge labeled "OwnedBy" between the TeamChatSession node and the Team node. Copy this script, run it in Neo4j, and verify that these nodes and edges are created.

iii. Screenshot of part of the graph:



Finding the longest conversation chain and its participants

A: First, find out what is the longest path length of the conversation.

match p=(m:ChatItem)-[:ResponseTo*]->(n:ChatItem) where m<>n return length(p) order by length(p) desc limit 1
Output: 9
Second, find out who are the participants in this longest conversation.
match p=(m:ChatItem)-[:ResponseTo*]->(n:ChatItem) where length(p)=9 with p match (u:User)-[r:CreateChat]->(i:ChatItem) where i in nodes(p) return count(distinct u)
Output: 5
The longest conversation chain has 9 edges and 5 users participating in it.

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

- First, find out the top 10 chattiest users:

  match (u:User)-[r:CreateChat]->(i:ChatItem) return u.id as user_id, count(r) as created_chat order by created_chat desc limit 10
- Second, find out the top 10 chattiest teams:

  match (i:ChatItem)-[r1:PartOf]->(c:TeamChatSession)-[r2:OwnedBy]->(t:Team) return t.id as team_id, count(r1) as team_chat order by team_chat desc limit 10
- Finally, find out in what the team the chattiest users joined

  match (u:User)-[r1:Join]->(c:TeamChatSession)-[r2:OwnedBy]->(t:Team) where u.id in [394, 2067, 209, 1087, 554, 1627, 516, 999, 668, 461] return distinct u.id, t.id

Chattiest Users

Users	Number of Chats
394	115
2067	111
1087	109

Chattiest Teams

Teams	Number of Chats
82	1324
185	1036
112	957

Among the top 10 chattiest users, only user with id 999 is in one of the top 10 chattiest team with team id 52. The other chattiest users are not in any of the top 10 chattiest teams.

How Active Are Groups of Users?

- First, create relationship between two users that the chat made by the first user is mentioned by the other user:

  match (u1:User)-[r1:CreateChat]->(i:ChatItem)-[r2:Mentioned]->(u2:User) where u1 <> u2 with u1, u2 create (u1)-[:InteractsWith]->(u2)
- Second, create relationship between two users that the first user made chat to respond to the chat which was made by the second user:

match (u1:User)-[r1:CreateChat]->(i1:ChatItem)-[r2:ResponseTo]->(i2: ChatItem)<-[r3:CreateChat]-(u2:User) where u1 <> u2 with u1, u2 create (u1)-[:InteractsWith]->(u2)

- Finally, calculate the coefficients of users:

match (u:User)-[:InteractsWith]-(n) where u.id in [394, 2067, 209, 1087, 554, 1627, 516, 999, 668, 461] with u, collect(distinct n) as neighbors, length(collect(distinct n))as neighbor_num match (a:User),(b:User) where (a in neighbors) and (b in neighbors) and (a <> b) return u.id, neighbor_num, (reduce(s=0,d in collect(case when (a)-->(b) then 1.0 else 0 end)|s+d))/(neighbor_num*(neighbor_num-1.0)) as clustering_coefficients order by clustering_coefficients desc

Most Active Users (based on Cluster Coefficients)

User ID	Coefficient
461	0.8333333333333334
209	0.8333333333333334
516	0.8095238095238095

Recommended Actions

Finally, based on our analysis with the data, in order to increase the revenue we recommend that:

1. The company could invest more in the game development on mobile ends including both android and iPhone, which contribute the most of the revenue.

2. Try to attract more iPhone users who are the major source of revenue in mobile ends.

3. Improve mission rewards, down-regulate mission difficulty to attract more casual players because most of the users are casual players. Thus, the company should take the casual players as the major target to grow the popularity.

4. Improve the quality of in-game ads to attract more clicks because the most spending users click ads very frequently. Better ads can arouse more of their interests so as to increase revenue.