Technical Appendix Catch the Pink Flamingo Analysis

Produced by: Metan KONE, **Capstone project, Big Data Specialization, offered by UC San Diego**

Note In this capstone project, we achieved the below analysis in the following order: ***Data Exploration , Classification, Clustering and Graph Analytics .***

1 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
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
game-clicks.csv	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
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.
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
users.csv	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	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.	timestamp: when the user joined the team. team: the id of the team userId: the id of the user assignmetId: 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	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
<Fill In>	<Fill in short phrase>	<Fill In: Name and describe all fields>

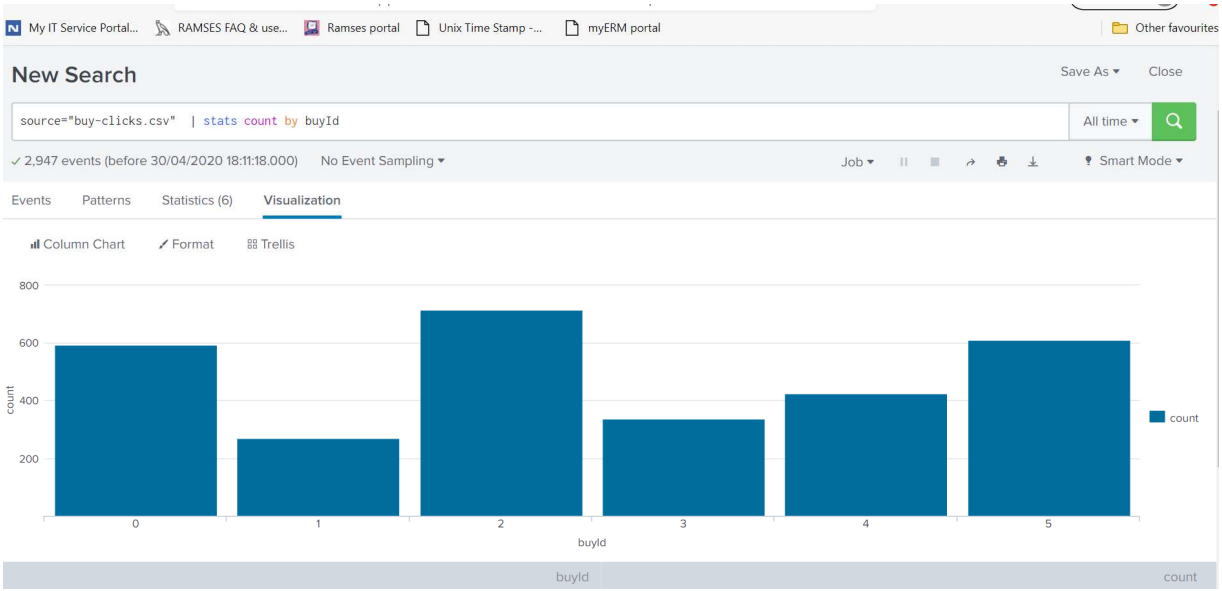
Files in green are the files actually used to provide answers bellow .

Aggregation

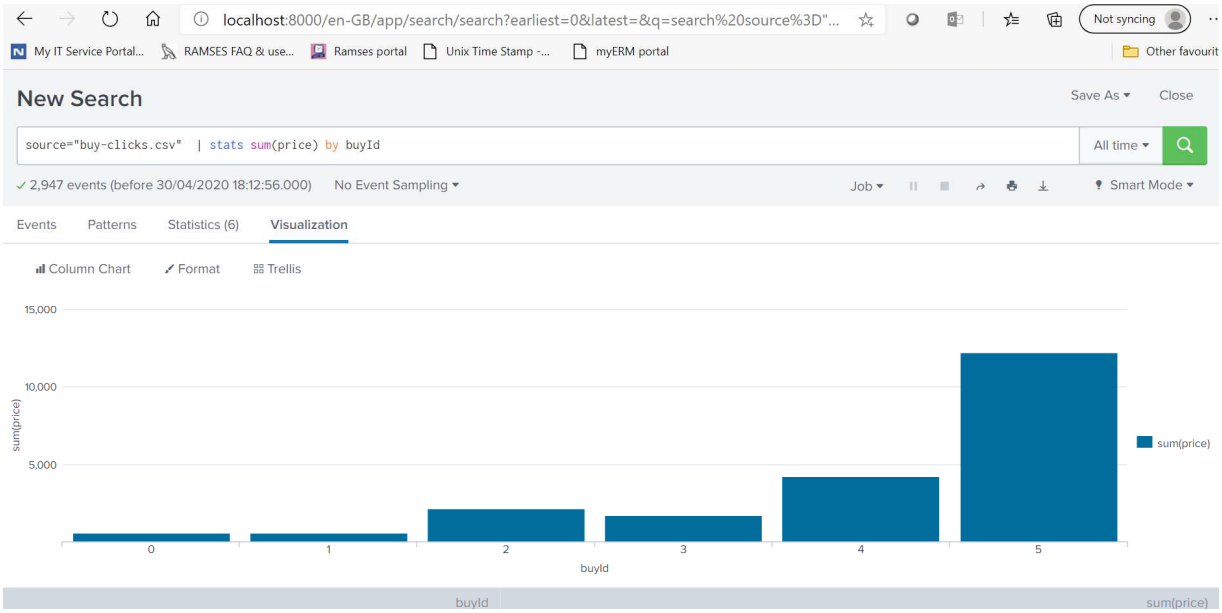
Amount spent buying items	21407
Number of unique items available to be purchased	6 unique items , (item ID , nbr of items purchased) 0 592 1 269 2 714 3 337 4 425 5 610

A histogram showing how many times each item is purchased:

<Fill In - upload screenshot>



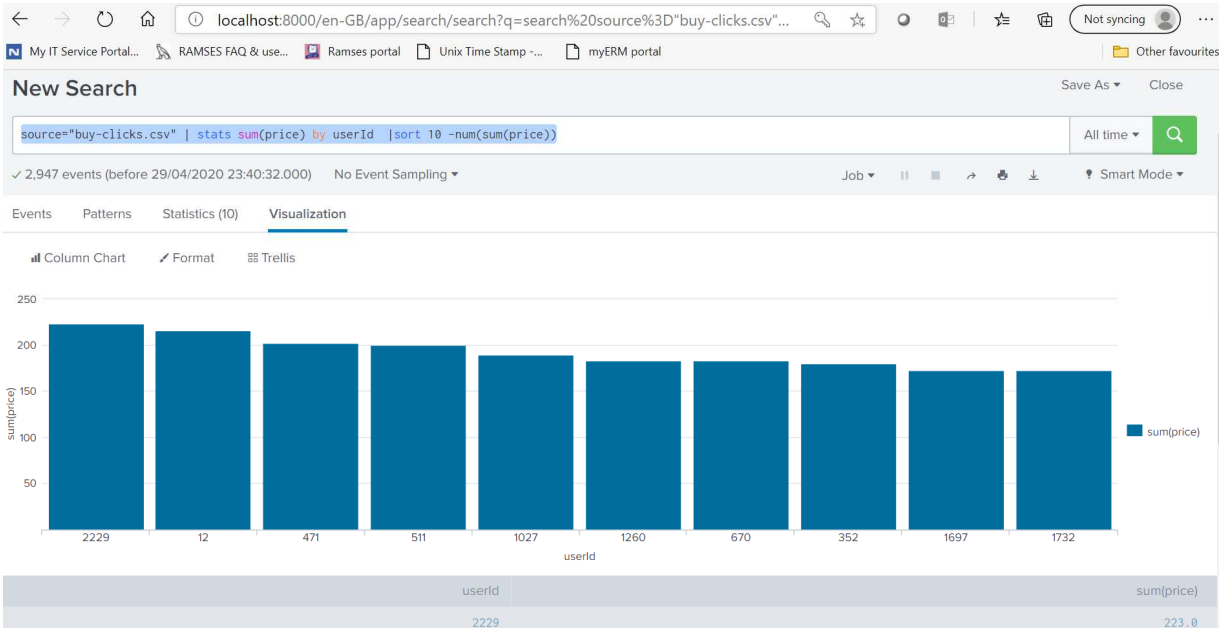
A histogram showing how much money was made from each item:



<Fill In - upload screenshot>

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	11.597
2	12	Iphone	13.07
3	471	Iphone	14.50

2 Classification

Part 1

Data Preparation

Analysis of combined_data.csv

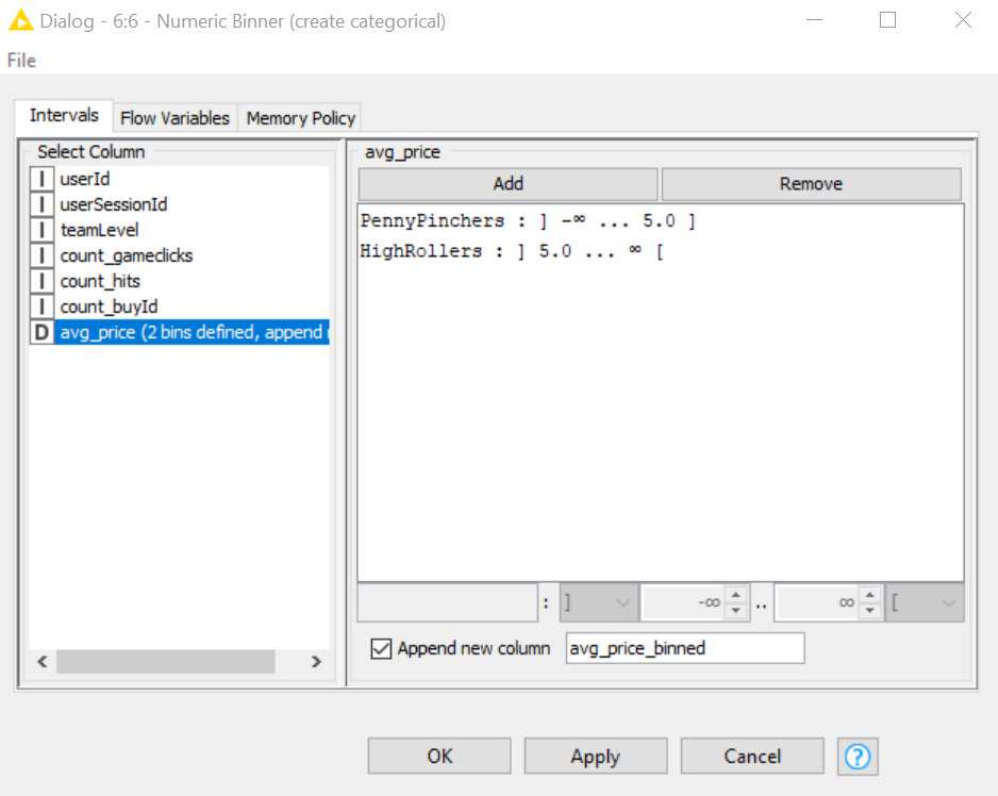
Sample Selection

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:

<Fill In Screenshot>



<Fill In: Describe the design of your attribute in 1-3 sentences.>

The new column is called avg_price_binned , and made of 2 categorical values : PennyPinchers paying less than $ 5 in average, and HighRollers paying more $ 5 in average.

We "Append new column" to use this new column in the flow.

The creation of this new categorical attribute was necessary because <Fill in 1-2 sentences>.

The goal of the present analytics exercise is to predict which users will fall HighRollers category , or in PennyPinchers category.

Attribute Selection

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

Attribute	**Rationale for Filtering**

avg_price	**Correlated with the new column avg_price_binned**
User_id	**No need, it's a number generated by the computer**
User_session	**No need, same reason like user_id**
~~<Optional Fill in>~~	~~<Optional Fill in 1-3 sentences>~~

Part 2

Data Partitioning and Modeling

The data was partitioned into train and test datasets.

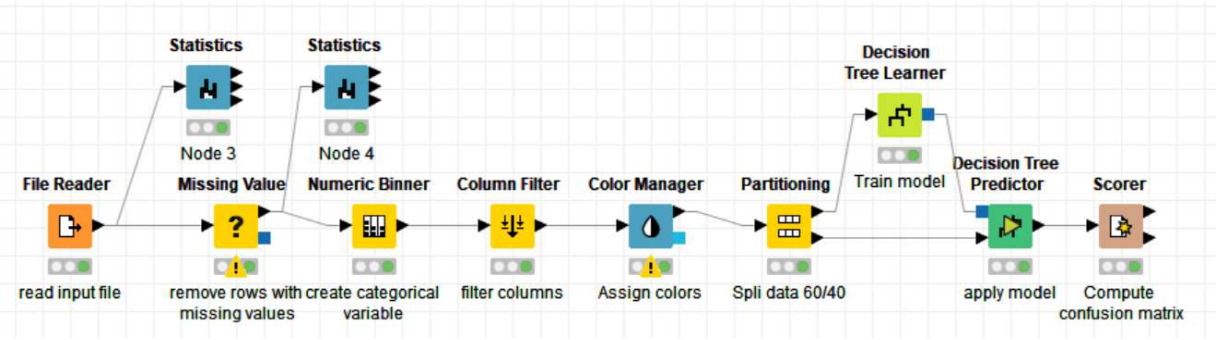
The **train** data set was used to create the decision tree model.

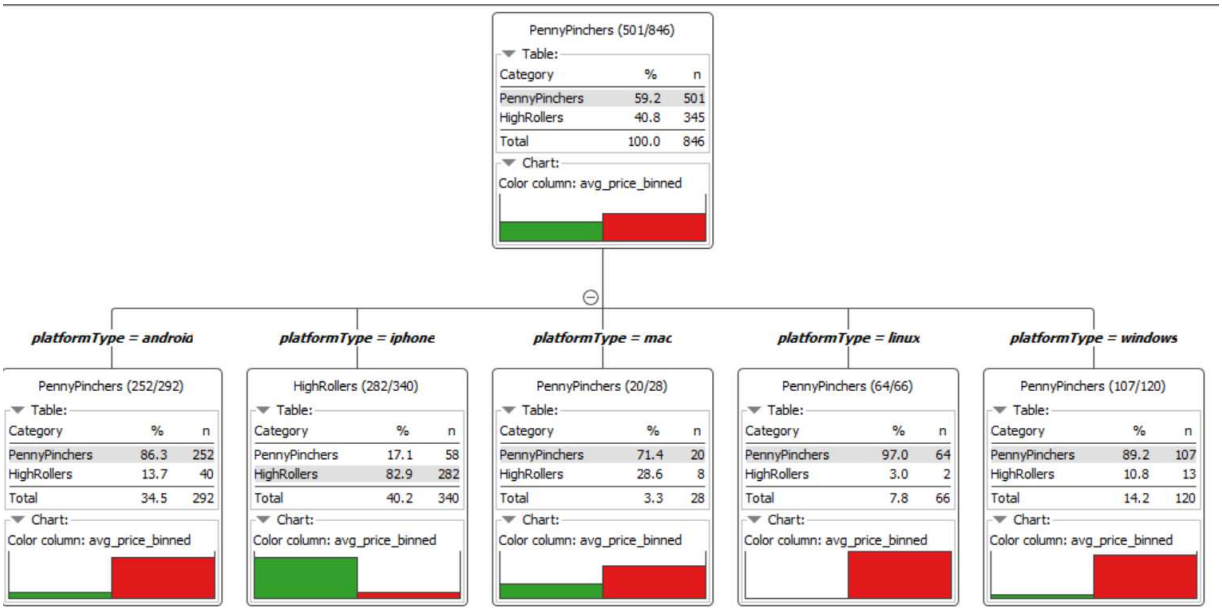
The trained model was then applied to the **test** dataset.

This is important because… **we must train the model with some data , and test the model with other data. We must not train the model and test it with the same data to avoid overfitting.**

When partitioning the data using sampling, it is important to set the random seed because… **we want repeatable result, meaning to get same result when repeating prediction with the same data. So we can compare our results and evaluate this exercise.**

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

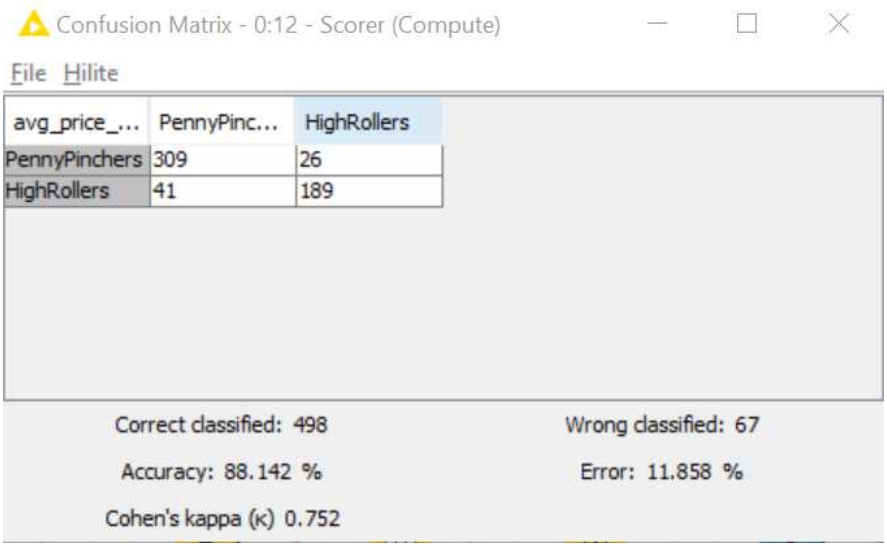


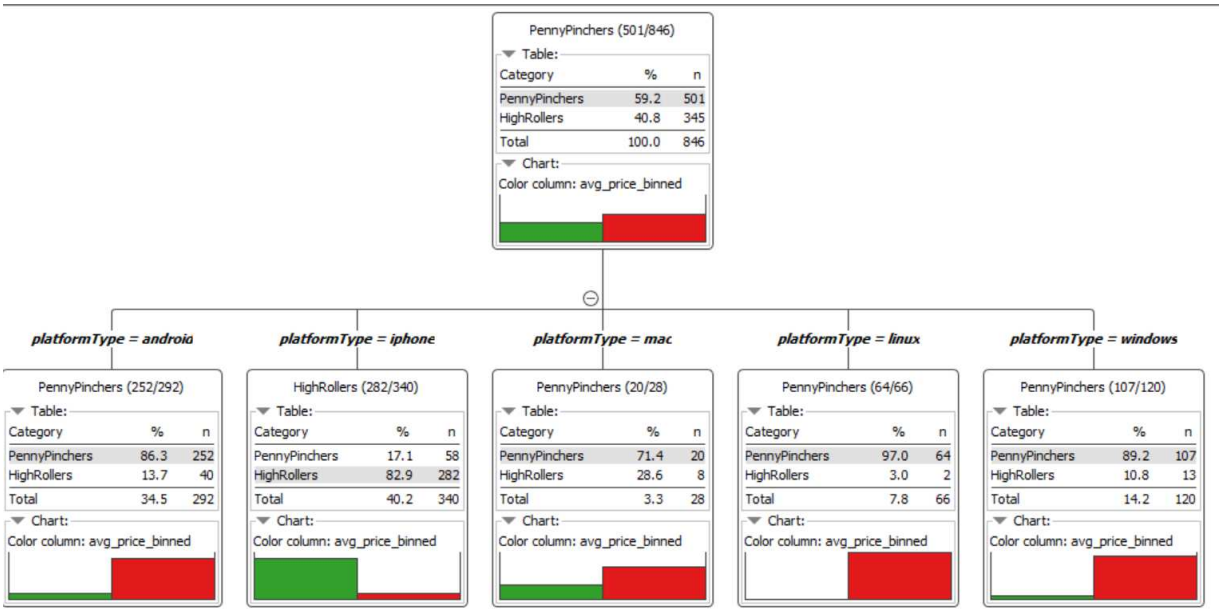


Part 3

Evaluation

A screenshot of the confusion matrix can be seen below:





As seen in the screenshot above, the overall accuracy of the model is **88.142%**

<Fill In: Write one sentence for each of the values of the confusion matrix indicating what has been correctly or incorrectly predicted.>

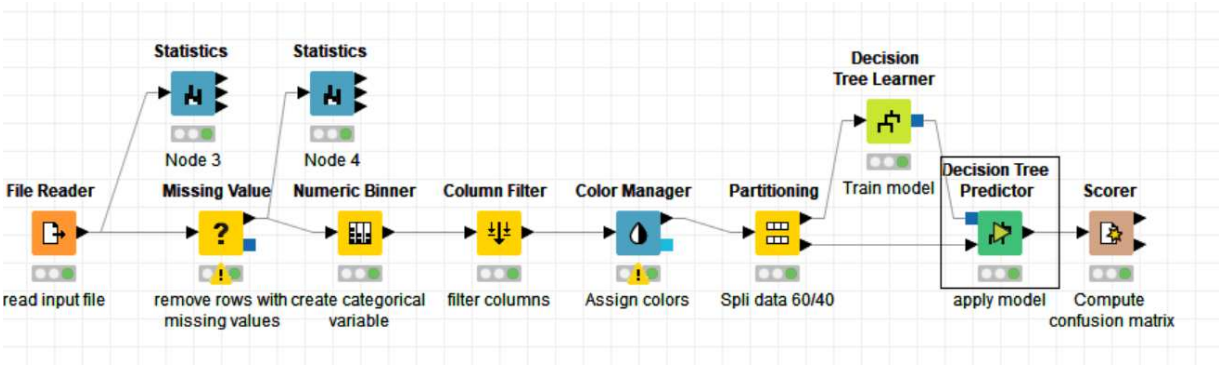
309 PennyPinchers were correctly classified, and 26 were wrongly classified.

189 HigRollers were correctly classified, and 41 were wrongly classified.

Part 4

Analysis Conclusions

The final KNIME workflow is shown below:



What makes a HighRoller vs. a PennyPincher?

<Fill In 2-3 sentences answering this question based on insights from your analysis.>

We notice players on iphone are high HighRollers. Other platforms players are PennyPichers.

User on Linux are high PennyPinchers.

Specific Recommendations to Increase Revenue
1. <Fill In 1-2 sentences describing 1 recommended action> **Show more advert to iphone users**.
2. <Fill In 1-2 sentences describing a 2nd recommended action> **Increase advert price for iphone platform.**

3 Clustering

Attribute Selection

features_used = [""]

Attribute	Rationale for Selection
count_gameclicks	Number of clicks done by players, will allow to identify player spending a lot of time .
~~count_hits~~	~~Will identifier best players, those ones will have high count_hits~~ (not used , as highly correlated to *hit_precision*)
hit_precision	Hit precision is a percentage of count_hits over all gamecliks
count_buyID	Number of items bought by players, will allow to identify those giving revenue to the company.
avg_price	Average amount of money spend buying on the game website.

Training Data Set Creation

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

```
In [123]: data.show()
```

```
+-----+---------------+----------+-----------+---------+-------------+
|RowID|count_gameclicks|count_hits|count_buyId|avg_price|hit_precision|
+-----+---------------+----------+-----------+---------+-------------+
|    1|             39|         0|          1|      1.0|          0.0|
|    2|            129|         9|          1|     10.0|  6.976744186|
|    3|            102|        14|          1|      5.0|   13.7254902|
|    4|             39|         4|          1|      3.0|  10.25641026|
|    5|             90|        10|          1|      3.0|  11.11111111|
|    6|             51|         8|          1|     20.0|  15.68627451|
|    7|             51|         6|          2|      2.5|  11.76470588|
|    8|             47|         5|          2|      2.0|  10.63829787|
|    9|             46|         7|          1|      1.0|   15.2173913|
|   10|             41|         6|          1|     20.0|  14.63414634|
|   11|             47|         7|          1|      3.0|  14.89361702|
|   12|             76|         7|          1|     20.0|  9.210526316|
|   13|             52|         2|          1|      3.0|  3.846153846|
|   14|             62|         9|          1|      3.0|  14.51612903|
|   15|            177|        25|          2|      7.5|  14.12429379|
|   16|             54|         5|          1|     10.0|  9.259259259|
|   17|             27|         4|          2|      4.0|  14.81481481|
|   18|             37|         2|          1|     20.0|  5.405405405|
|   19|             67|         5|          1|      1.0|  7.462686567|
|   20|             37|         5|          2|     11.5|  13.51351351|
+-----+---------------+----------+-----------+---------+-------------+
only showing top 20 rows
```

Dimensions of the final data set: **6 columns, 1411 rows**

# of clusters created: **5 clusters (after elbow analysis)**

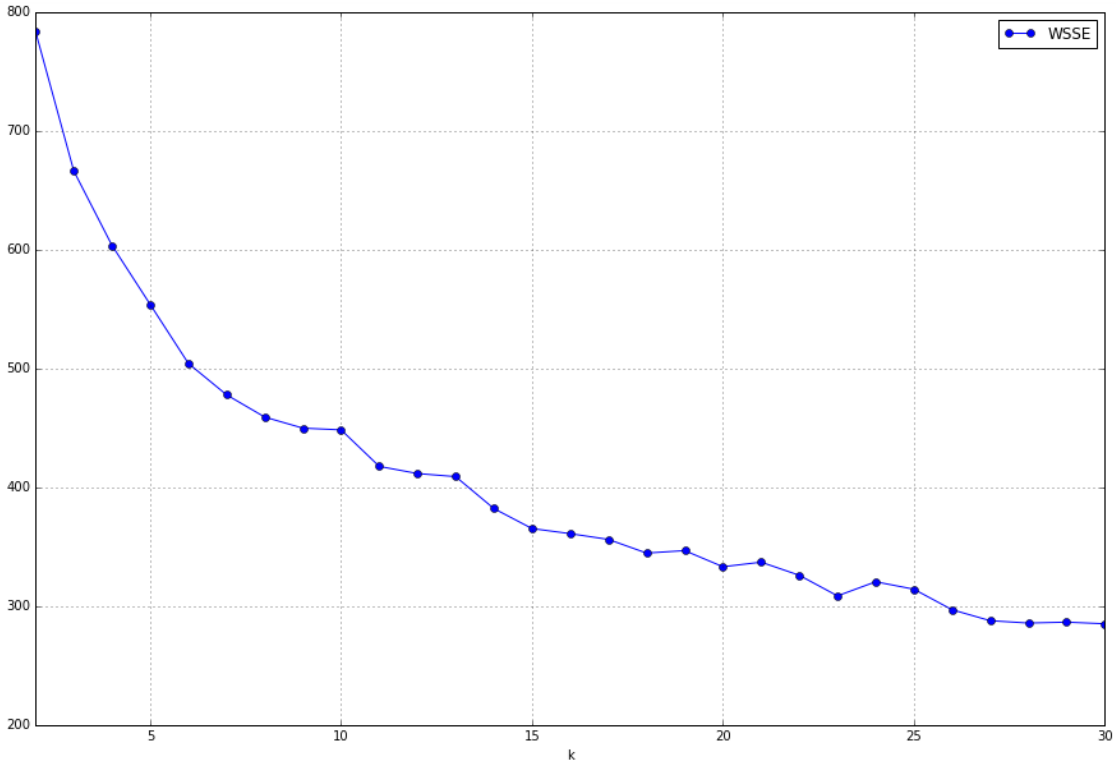
Cluster Centers

The code used in creating cluster centers is given below:

Creating WSSE for clusters having size 2 to 32.

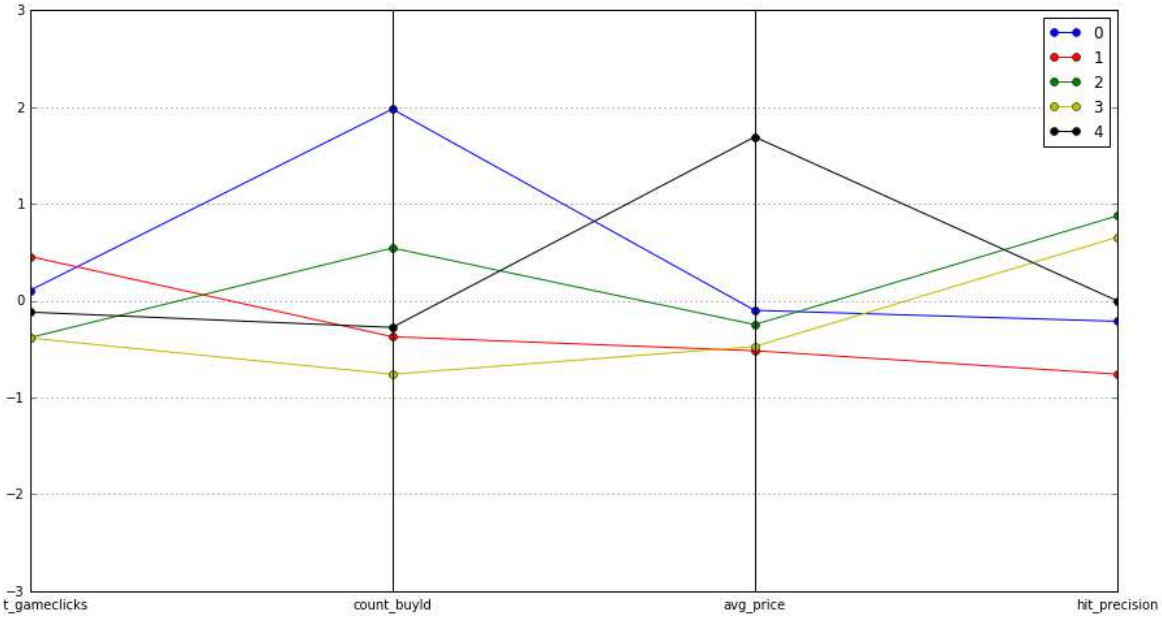
```
In [12]: featuresUsed = ['count_gameclicks','count_buyId','avg_price','hit_precision']
         assembler = VectorAssembler(inputCols=featuresUsed, outputCol="features_unscaled")
         assembled = assembler.transform(data)

In [13]: scaler = StandardScaler(inputCol="features_unscaled", outputCol="features", withStd=True, withMean=True)
         scalerModel = scaler.fit(assembled)
         scaledData = scalerModel.transform(assembled)

In [14]: scaledData = scaledData.select("features", "RowID")
         elbowset = scaledData.filter((scaledData.RowID % 3) ==0).select("features")
         elbowset.persist()
Out[14]: DataFrame[features: vector]

In [15]: clusters = range(2,31)
         wsseList = utils.elbow(elbowset, clusters)

         Training for cluster size 2
         ......................WSSE = 783.1945935636946
         Training for cluster size 3
         ......................WSSE = 666.0651900767973
         Training for cluster size 4
         ......................WSSE = 603.5000345708783
         Training for cluster size 5
```

Elbow method shows 10 is a good number of clusters to use :



Cluster centers formed are given in the table below

Cluster #	Center
0	0.10723134, 1.980912778, -0.09852266, -0.21182086
1	0.45816666,-0.37126505, -0.51606561, -0.75776555
2	-0.38066003, 0.54586233, -0.24610162, 0.87695128
3	-0.38471387, -0.75868591, -0.4727514, 0.65651244
4	-1.17148452e-01, -2.74952690e-01, 1.69343828e+00, 1.03948322e-03



These clusters can be differentiated from each other as follows:

Cluster 0 has the highest number count_buyID, but we can notice the revenue is coming most from cluster 4.

Cluster 1 members are most playing . We can therefore notice they have the less precision. Also , Cluster 1 members spend a lot of time playing but they are NOT giving revenue.

Cluster 3 members who plays in a moderate way , have more hit precision .

Below you can see the summary of the train data set:

<SCREENSHOT AS FROM JUPYTER NOTEBOOK>

```
In [21]: kmeans = KMeans(k=5, seed=1)
         model = kmeans.fit(scaledData)
         transformed = model.transform(scaledData)

In [22]: centers = model.clusterCenters()
         centers

Out[22]: [array([ 0.10723134,  1.98091278, -0.09852266, -0.21182086]),
          array([ 0.45816666, -0.37136505, -0.51606561, -0.75776555]),
          array([-0.38066003,  0.54586233, -0.24610162,  0.87695128]),
          array([-0.38471387, -0.75868591, -0.4727514 ,  0.65651244]),
          array([ -1.17148452e-01,  -2.74952690e-01,   1.69343828e+00,
                   1.03948322e-03])]
```

Recommended Actions

Action Recommended	**Rationale for the action**
Increase ads to users who play a lot	It was seen that users who play a lot are also the users who spend less and click less on ads. If we increase ads to users who play a lot, it will promote these users to spend more and therefore increase the revenue. **As they play a lot of , "reward " them with discount so they will get used to to buy .**
Show higher price ads to users who spend more	If we show higher price ads to users who spend more, we can increase the revenue faster.The users who spend the more also do not play too much, thus by showing them the more valuable ads first, we can increase the revenue faster
Show best tips and techniques to players spending time playing.	**To allow them have more precision , and enjoy gaming , so they will keep playing , and buying (if they are rewarded with discounts, they will get use to buying items)**

4. Graph Analytics

Modeling Chat Data using a Graph Data Model

The chat data defines 4 nodes (User, Team, TeamChatSession, ChatItem) and following edges CreatesSession, OwnedBy, Joins, Leaves, CreateChat, part of, Mentioned, and InteractsWith.

Nodes are using properties "id" while the edges use timestamp as property to define when the actions were done.

Creation of the Graph Database for Chats

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

i) Write the schema of the 6 CSV files

chat_create_team_chat.csv :
column 1 : User {id: Integer}
column 2 : Team {id: Integer}
column 3 : TeamChatSession {id: Integer}
column 4 : CreatesSession {timestamp}
column 5 : OwnedBy{timestamp}

chat_join_team_chat.csv :
column 1 : User {id: Integer}
column 2 : TeamChatSession {id: Integer}
column 3 : Joins{timestamp}

chat_leave_team_chat.csv :
column 1 : User {id: Integer}
column 2 : TeamChatSession {id: Integer}
column 3 : Leaves{timestamp}

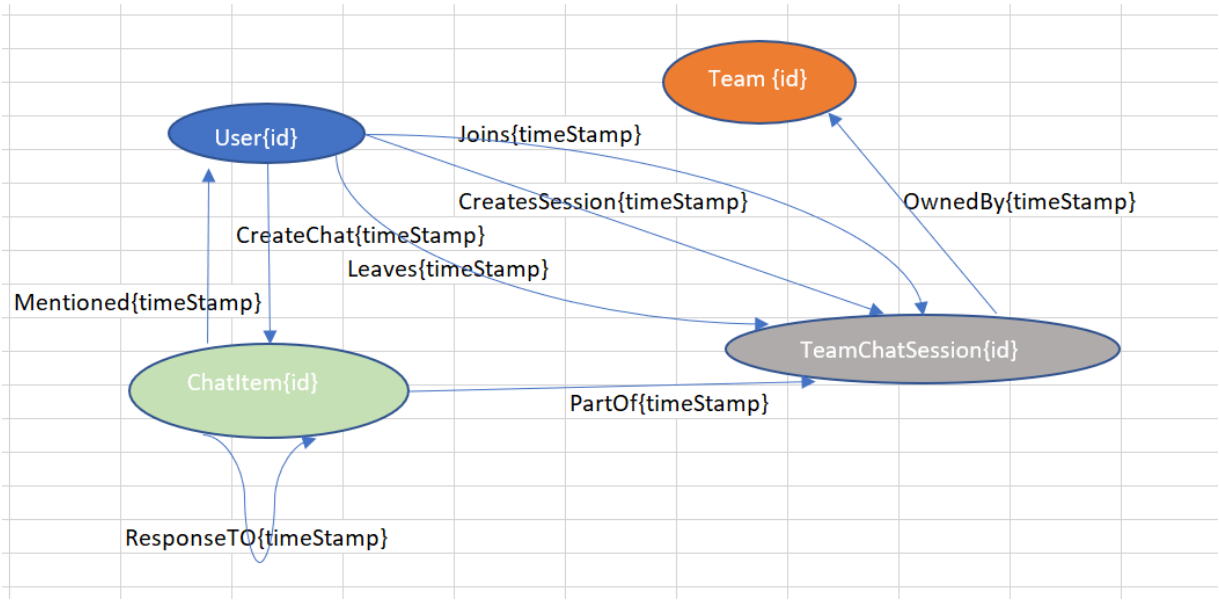
chat_item_team_chat.csv :
column 1 : User {id: Integer}
column 2 : TeamChatSession {id: Integer}
column 3 : ChatItem {id: toInteger}
column 4 : CreateChat{timestamp} and PartOf{timestamp}

chat_mention_team_chat.csv :
column 1 : ChatItem {id: Integer}
column 2 : User {id: Integer}

column 3 : Mentioned{timestamp}

chat_respond_team_chat.csv :

column 1 : ChatItem {id: Integer}
column 2 : ChatItem {id: Integer}
column 3 : ResponseTO {timeStamp}

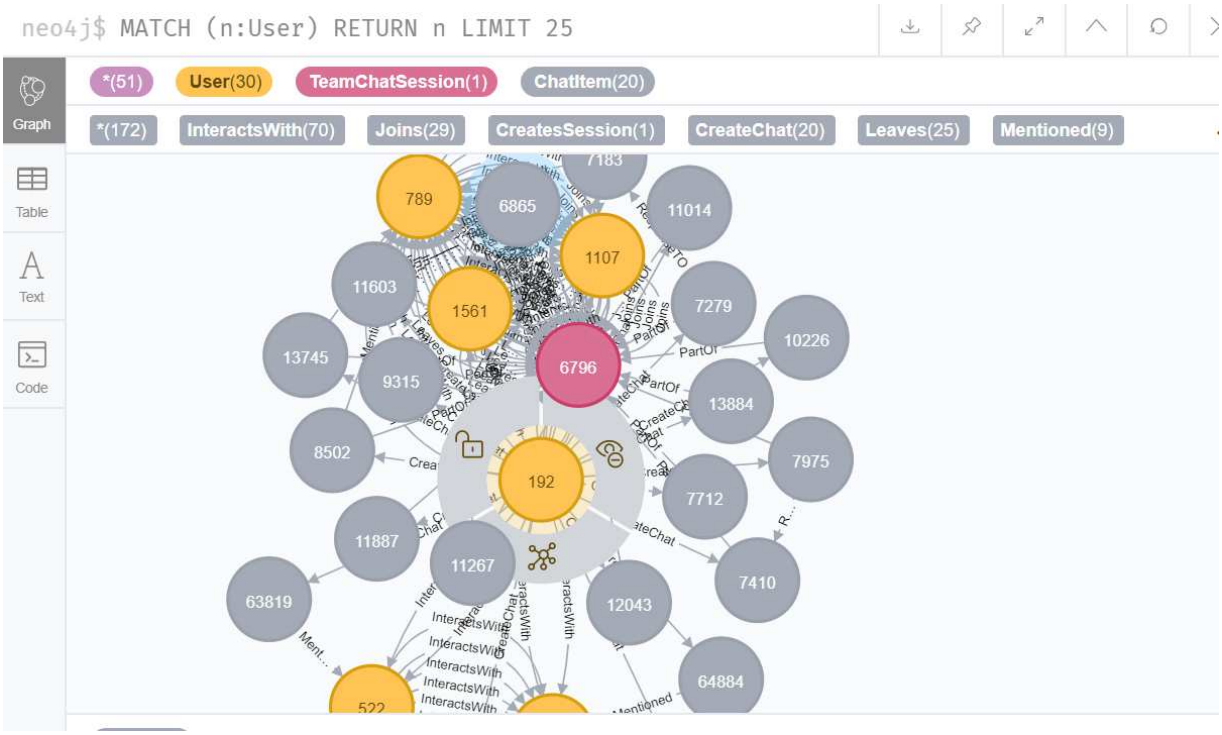


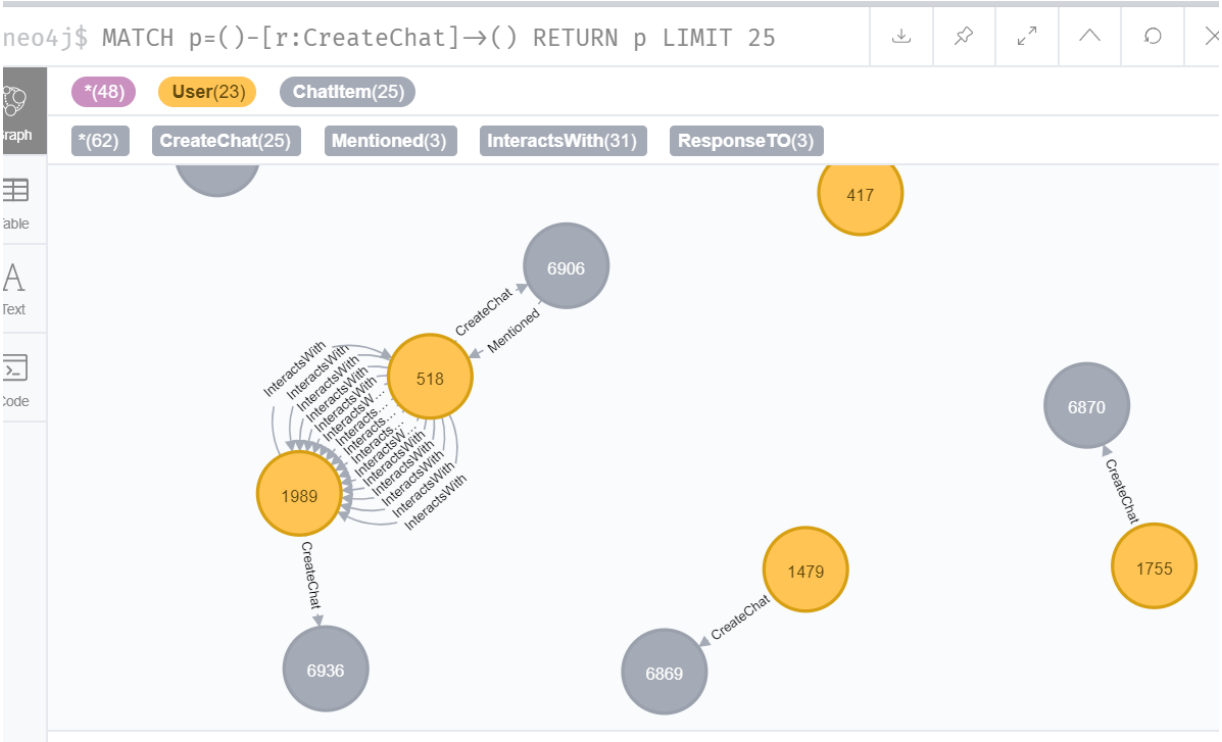
ii) Explain the loading process and include a sample LOAD command

1st line of command below specify the path of the CSV file to load . The CSV file contains 3 columns:
1st column is loaded as User on 2nd line of code, 2nd column is loaded as TeamChatSession at 3rd line . 3rd column contains timestamp is loaded at line 4.

```
LOAD CSV FROM "file:///C:chat_data/chat_join_team_chat.csv" AS row
MERGE (u:User {id: toInteger(row[0])})
MERGE (c:TeamChatSession {id: toInteger(row[1])})
MERGE (u)-[:Joins{timeStamp: row[2]}]->(c)
```

iii) Present a screenshot of some part of the graph you have generated. The graphs must include clearly visible examples of most node and edge types. Below are two acceptable examples. The first example is a rendered in the default Neo4j distribution, the second has had some nodes moved to expose the edges more clearly. Both include examples of most node and edge types.


neo4j$ MATCH (n:User) RETURN n LIMIT 25
Graph
Table
Text
Code
*(51)
User(30)
TeamChatSession(1)
ChatItem(20)
*(172)
InteractsWith(70)
Joins(29)
CreatesSession(1)
CreateChat(20)
Leaves(25)
Mentioned(9)
789
6865
7183
11014
1107
11603
1561
7279
10226
13745
9315
6796
13884
8502
192
7975
7712
11887
11267
7410
63819
12043
64884
522
PartOf
InteractsWith
CreateChat
Mentioned

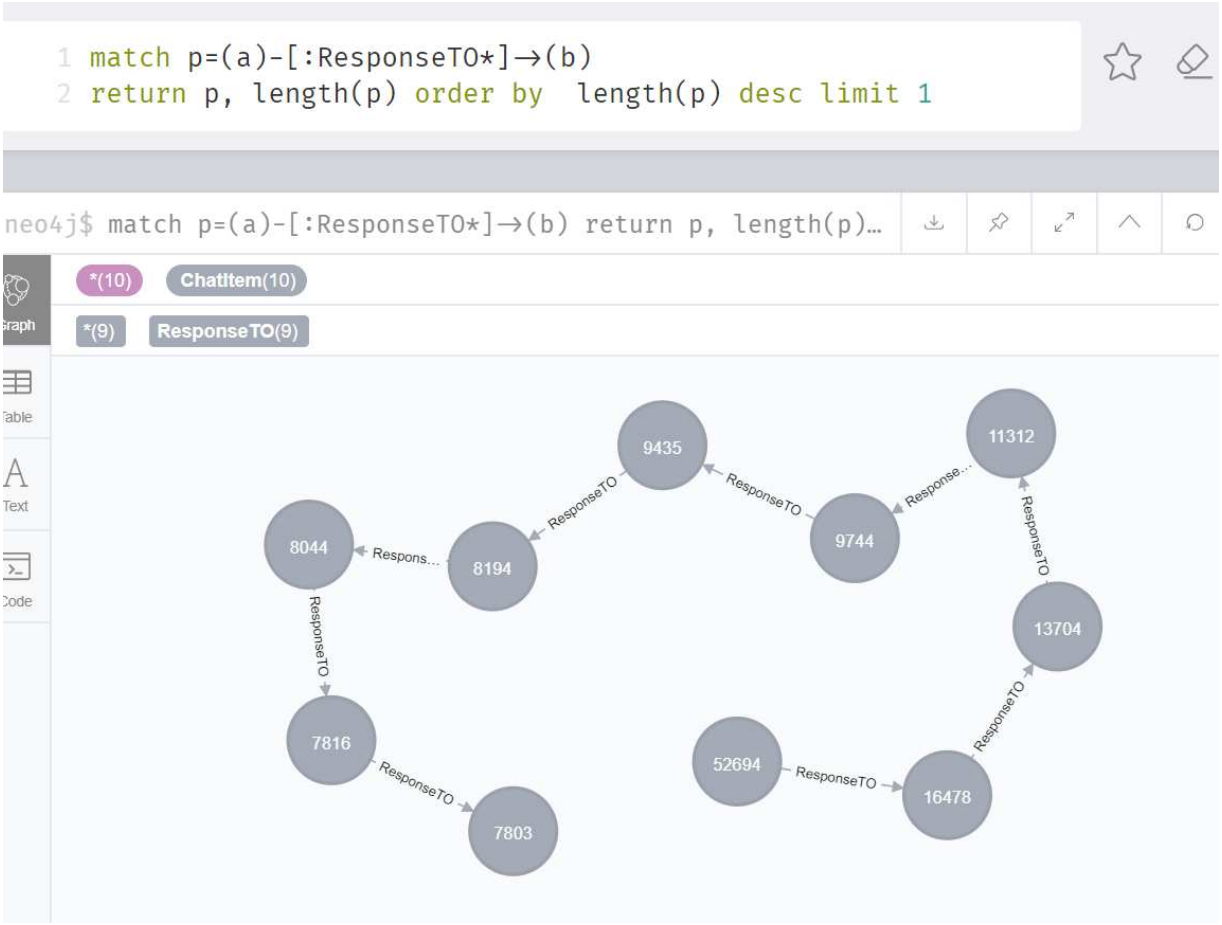

neo4j$ MATCH p=()-[r:CreateChat]→() RETURN p LIMIT 25
*(48)
User(23)
ChatItem(25)
*(62)
CreateChat(25)
Mentioned(3)
InteractsWith(31)
ResponseTO(3)
417
6906
CreateChat
Mentioned
518
InteractsWith
1989
6870
CreateChat
1755
1479
CreateChat
6869
CreateChat
6936

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.

```
match p=(a)-[:ResponseTO*]->(b)
return p, length(p) order by  length(p) desc limit 1
```

=> 9 Chats involved in the longest conversation;



5 unique users in this conversation:

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

Describe your steps from Question 2. In the process, create the following two tables. You only need to include the top 3 for each table. Identify and report whether any of the chattiest users were part of any of the chattiest teams.

Chattiest Users

Users	Number of Chats
394	**115**
2067	**111**
1087	**109**

Chattiest Teams

Teams	Number of Chats
82	**1324**
185	**1036**

112	**957**

Finally, present your answer, i.e. whether or not any of the chattiest users are part of any of the chattiest teams.

YES, User 999 that is part of top 10 chattiest users, belong to team 52 that is part of the top 10 chattiest team. We find all the users who joins a chatSession and corresponding Teams that own those TeamChatSession; Among those users, we search by User.id (top 10 chattiest user id) , and we can find their corresponding teams. Only user 999 teams is present in the top 10 chattiest team (team 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.

The answer of this question is based on the new edges created (InteractsWith) to estimate of how "dense" the neighborhood of a node is.

A clustering coefficient is a score ranging from 0 (a disconnected user) to 1. For example, if the number of neighbors of a node is 5, then the clustering coefficient of the node is the ratio between the number of edges amongst these 5 neighbors(not counting the given node) and 5 * 4 / 2 (all the pairwise edges that could possibly exist). Thus the denominator is k * (k-1) / 2 if the number of neighbors of the node is k.

We get the list of neighbors and the number of neighbors of a node based on the "InteractsWith" edge.

Most Active Users (based on Cluster Coefficients)

User ID	Coefficient
209	0.952
554	0.90
1087	0.8