

Technical Appendix

Catch the Pink Flamingo Analysis

Produced by: Nima Afshari

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.	<p>timestamp: when the purchase was made.</p> <p>txId: a unique id (within buy-clicks.log) for the purchase</p> <p>userSessionId: the id of the user session for the user who made the purchase</p> <p>team: the current team id of the user who made the purchase</p> <p>userId: the user id of the user who made the purchase</p> <p>buyId: the id of the item purchased</p> <p>price: the price of the item purchased</p>
users.csv	This file contains a line for each user playing the game.	<p>timestamp: when user first played the game.</p> <p>userId: the user id assigned to the user.</p> <p>nick: the nickname chosen by the user.</p> <p>twitter: the twitter handle of the user.</p> <p>dob: the date of birth of the user.</p>

		country: the two-letter country code where the user lives.
team.csv	This file contains a line for each team terminated in the game.	<p>teamId: the id of the team</p> <p>name: the name of the team</p> <p>teamCreationTime: the timestamp when the team was created</p> <p>teamEndTime: the timestamp when the last member left the team</p> <p>strength: a measure of team strength, roughly corresponding to the success of a team</p> <p>currentLevel: the current level of the team</p>
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.	<p>timestamp: when the user joined the team.</p> <p>team: the id of the team</p> <p>userId: the id of the user</p> <p>assignmentId: a unique id for this assignment</p>

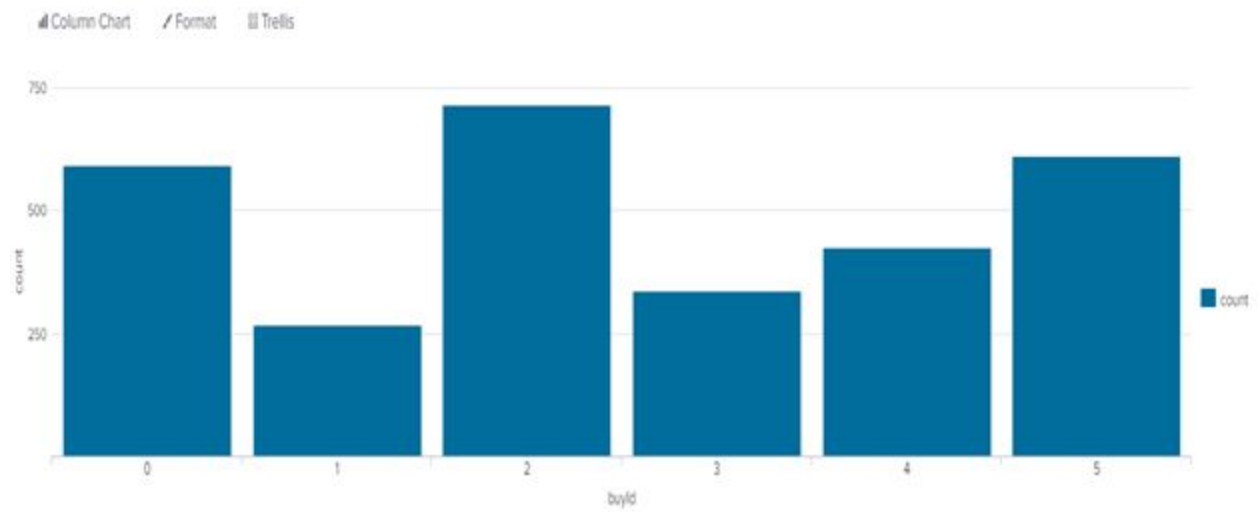
level-events.csv	A line is added to this file each time a team starts or finishes a level in the game	<p>timestamp: when the event occurred.</p> <p>eventId: a unique id for the event</p> <p>teamId: the id of the team</p> <p>teamLevel: the level started or completed</p> <p>eventType: the type of event, either start or end</p>
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.	<p>timestamp: a timestamp denoting when the event occurred.</p> <p>userSessionId: a unique id for the session.</p> <p>userId: the current user's ID.</p> <p>teamId: the current user's team.</p> <p>assignmentId: the team assignment id for the user to the team.</p> <p>sessionType: whether the event is the start or end of a session.</p> <p>teamLevel: the level of the team during this session.</p> <p>platformType: the type of platform of the user during this session.</p>

game-clicks.csv	A line is added to this file each time a user performs a click in the game.	<p>timestamp: when the click occurred.</p> <p>clickId: a unique id for the click.</p> <p>userId: the id of the user performing the click.</p> <p>userSessionId: the id of the session of the user when the click is performed.</p> <p>isHit: denotes if the click was on a flamingo (value is 1) or missed the flamingo (value is 0)</p> <p>teamId: the id of the team of the user</p> <p>teamLevel: the current level of the team of the user</p>
------------------------	---	--

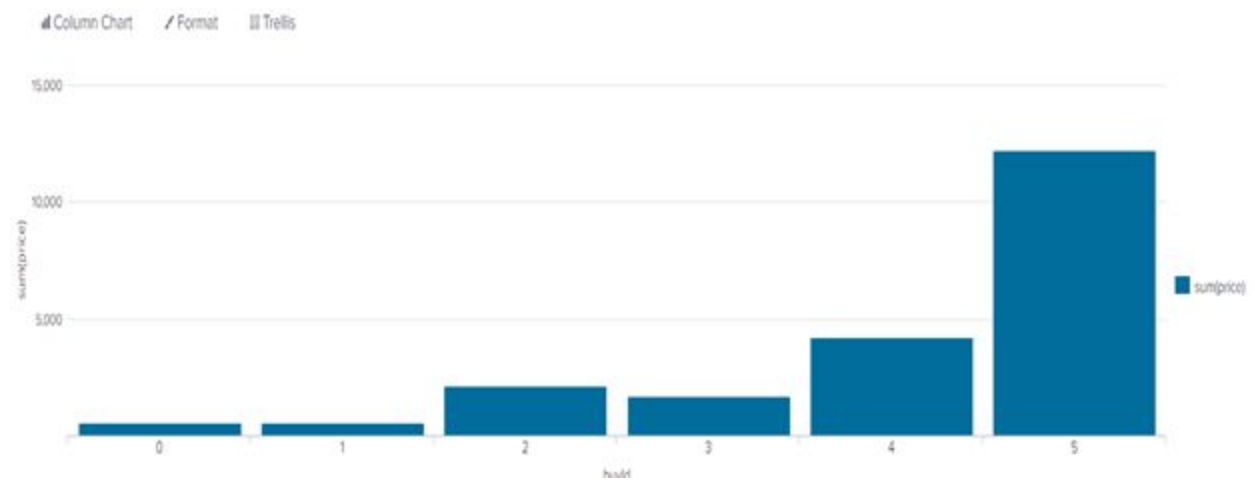
Aggregation

Amount spent buying items	21407.0
Number of unique items available to be purchased	6

A histogram showing how many times each item is purchased:

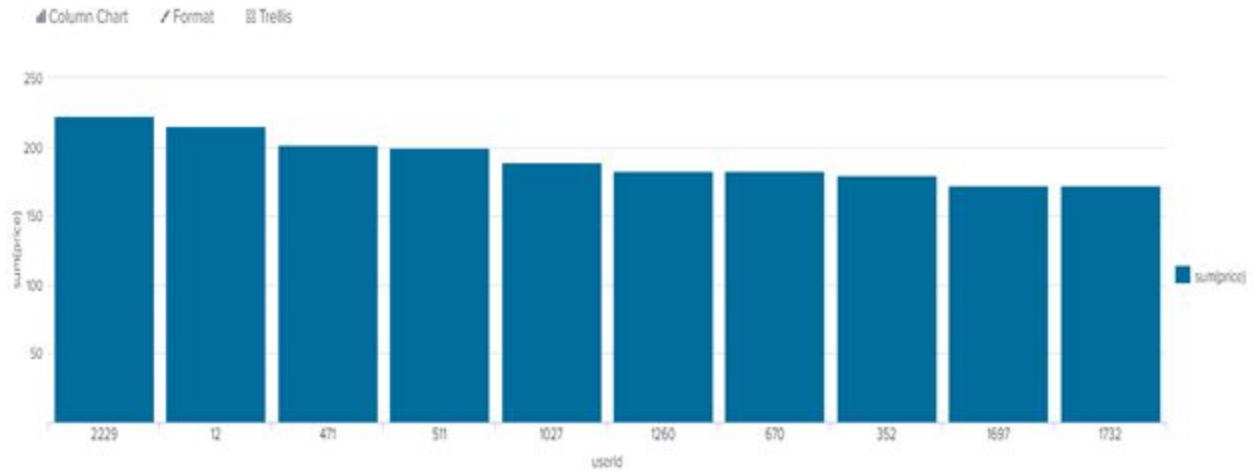


A histogram showing how much money was made from each item:



Filtering

A histogram showing the 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	11.58
2	12	iphone	13.08
3	471	iphone	14.50

Data Classification Analysis

Data Preparation

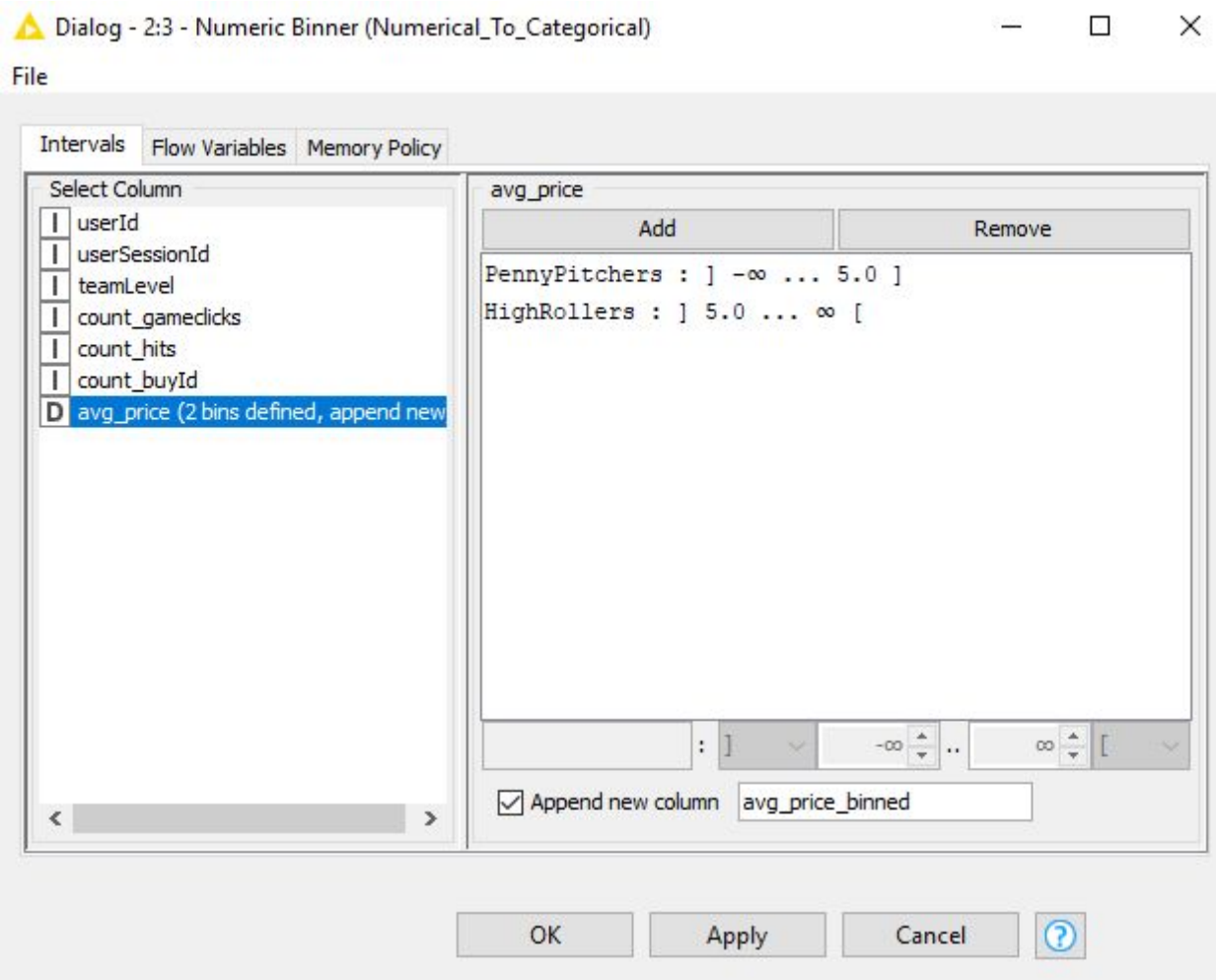
Analysis of combined_data.csv

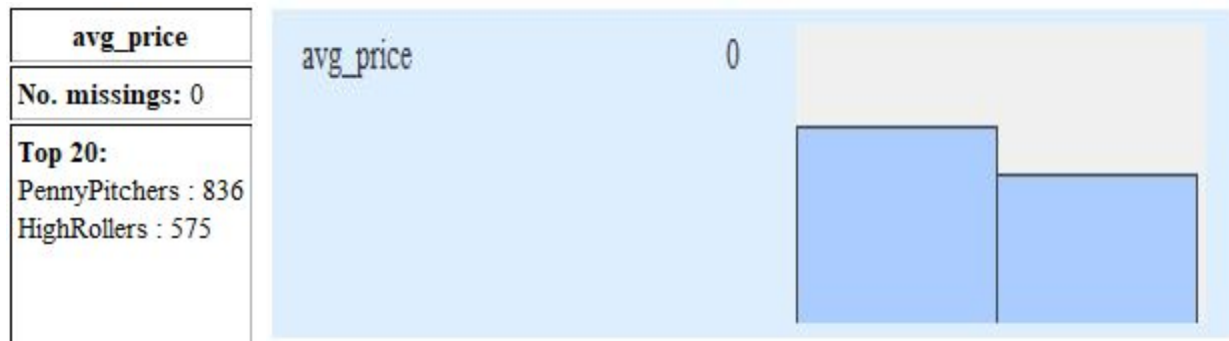
Sample Selection

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

Attribute Creation

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





According to the Eglence needs the classes would be the users that purchased items with a price more than 5 USD and equal or below 5 USD. These groups are called PennyPitchers and HighRollers classes and the classification goal is to properly allocate users into these two groups.

Each classification problem requires a discrete number of classes. This is in contrast to the regression that can have a continuous variable as the target. Thus, a transformation from the price that is continuous to a two class discrete variable has been done based on customer needs to assign each user based on their activity.

Attribute Selection

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

Attribute	Rationale for Filtering
userId	Obviously, a randomly generated userId cannot be related to the purchasing decisions that a user make. This property is normally used as a primary key for storing and retrieval of each entry in a DBMS or spreadsheet. Hence, it can be safely removed.
userSessionId	Just like userId, a userSessionId is another randomly generated number to separate various sessions for a user. Thus, that couldn't have any logical relation to the user purchasing process. This is removed from the attributes as well.
avg_price	This attribute is not needed anymore, as it is replaced with another categorical one.

Data Partitioning and Modeling

The data was partitioned into train and test datasets.

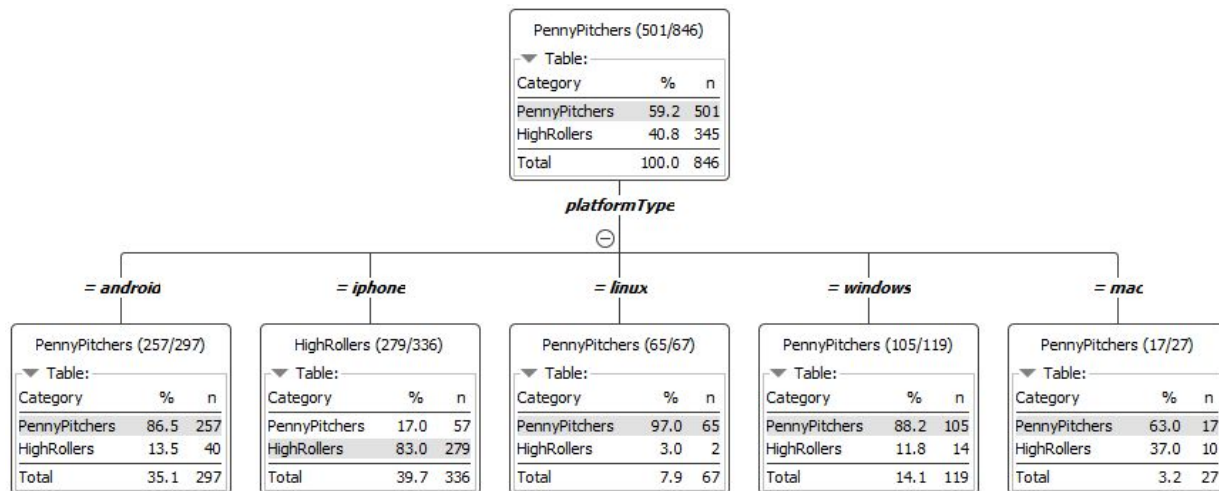
The train data set was used to create a decision tree model.

The trained model was then applied to the test dataset.

This is important because the classifier could overfit to the training data and cannot decide about new inputs' classes. Thus, this test set is used to check if the classifier is powerful enough to make correct predictions on new data.

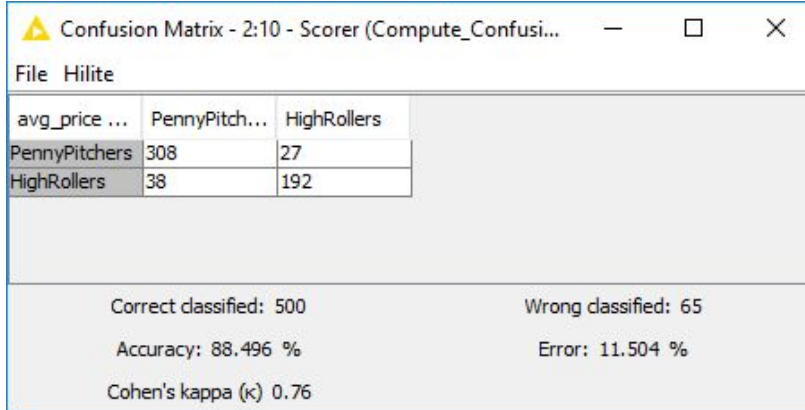
When partitioning the data using sampling, it is important to set the random seed because of reproducibility purpose. To make a random number generator always make the same set of numbers the seed variable should be set.

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



Evaluation

A screenshot of the confusion matrix can be seen below:



The screenshot shows a window titled "Confusion Matrix - 2:10 - Scorer (Compute_Confusi...)" with a menu bar containing "File" and "Hilite". It displays a confusion matrix for two classes: PennyPitchers and HighRollers. The matrix shows 308 correct PennyPitchers, 27 PennyPitchers misclassified as HighRollers, 38 HighRollers misclassified as PennyPitchers, and 192 correct HighRollers. Below the matrix, summary statistics are provided: Correct classified: 500, Wrong classified: 65, Accuracy: 88.496 %, Error: 11.504 %, and Cohen's kappa (κ) 0.76.

	PennyPitch...	HighRollers
PennyPitchers	308	27
HighRollers	38	192

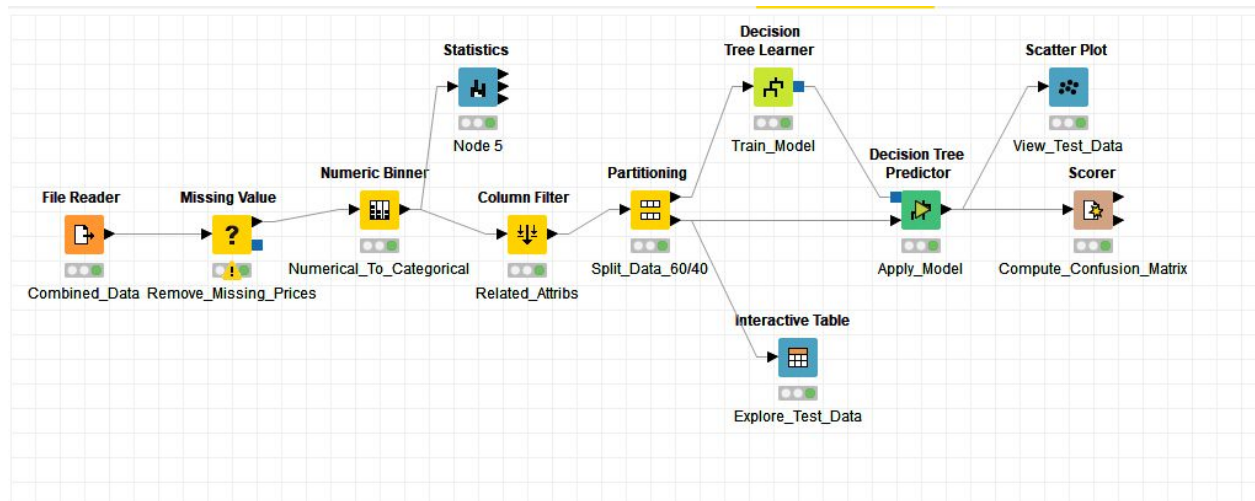
Correct classified: 500 Wrong classified: 65
Accuracy: 88.496 % Error: 11.504 %
Cohen's kappa (κ) 0.76

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

Add to that, it is clear that the model has predicted 27 of PennyPitchers as HighRollers, and 38 of HighRollers as PennyPitchers. Rather than that other samples in the test set has been predicted correctly to their class.

Analysis Conclusions

The final KNIME workflow is shown below:



What makes a HighRoller vs. a PennyPincher?

The decision tree classified the users based on the operating systems, using Gini Index as an indicator of the best attribute. Looking closer look to the resulting tree, it is pretty obvious that more than the iPhone users are HighRollers. On the other hand, the users of other operating systems are mostly PennyPitchers. Thus, it concludes the operating system of a new user can be a suitable indicator of their purchases in the future. This result seems logical based on that iPhone users are those people that spend more on technology and virtual commodities.

Specific Recommendations to Increase Revenue
1. Put some effort on Native iPhone apps and make the app faster and more fancy to make the users attracted to spend more time in it. Spending more time in the app can lead to engagement and hence more purchasing.
2. Start some sales funnels specifically for iPhone users to elevate the number of them. This would lead to more HighRoller users and more revenue for the company.

Clustering Analysis

Attribute Selection

Attribute	Rationale for Selection
Total number of ad-clicks per user	This feature distinguishes users that pay attention to in-app ads and those users that just close the ads after they pop up. It can be a rough sign of a curious personality of the users that also can make money for the company. These users are the target for money making ad campaigns.
Some of the money spent on all ad categories by each user	This is a good feature to show how the ad campaigns are working. This attribute is different from in-app purchases.
Game clicks per hour by each user	This feature is a measure of spending time and user interaction with the app. That is the mean of clicks by each user on a window of one hour. It shows another aspect of users' behavior.

Training Data Set Creation

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

```
[Row(avg_game_clicks_per_hour=2.3248407643312103, Total_ad_Clicks=19, Total_buy($)=63.0),  
Row(avg_game_clicks_per_hour=2.051051051051051, Total_ad_Clicks=39, Total_buy($)=20.0),  
Row(avg_game_clicks_per_hour=1.3732057416267942, Total_ad_Clicks=37, Total_buy($)=28.0),  
Row(avg_game_clicks_per_hour=2.629139072847682, Total_ad_Clicks=34, Total_buy($)=95.0),  
Row(avg_game_clicks_per_hour=1.9754601226993864, Total_ad_Clicks=41, Total_buy($)=53.0)]
```

Dimensions of the training data set (rows x columns) : (543 , 3)

of clusters created: 3

Cluster Centers

Cluster #	Center
1	[0.34491307, -0.88428349, -0.4376942]
2	[-0.21638267, 0.72866978, 2.00843352]
3	[-0.34502313, 0.8193491, -0.21235772]

These clusters can be differentiated from each other as follows:

Cluster 1 is different from the others in that users in this cluster are those that are spending the longest time in the game, but rarely click on ads or buy something. It seems that the cluster is not useful for a direct planned marketing or purchasing. However, they are the best to expand the app with their friends and the company can plan on incentives for them.

Cluster 2 is different from the others in that they are affluent ones and at the same time curious about ads. They like to buy online materials and they are spending money on the ads and in-app purchases. Targeted sales funnels will work best for this cluster users.

Cluster 3 is different from the others in that they just click on ads but not spending money in the app. Thus, the company should make some in-game incentives for these users to attract them for clicking more on the money-making ads.

Recommended Actions

Action Recommended	The rationale for the action
Cluster 1	they are the best to expand the app with their friends and the company can plan on incentives for them. Using any sort of marketing with this goal seems reasonable.
Cluster 2	They like to buy online materials and they are spending money on the ads and in-app purchases. Targeted sales funnels will work best for this cluster users.
Cluster 3	The company should make some in-game incentives for these users to attract them for clicking more on the money-making ads.

Graph Analytics

Modeling Chat Data using a Graph Data Model

Basically, when there are many nested relationships between database entities, it is more proper to use a graph database instead of an RDBMS. That would be a great help for querying the data in the future. The generated graph here contains Nodes (User, Team, ChatItem, TeamChatSession), and there are Edges (CreateChat, CreatesSession, Joins, Leaves, Mentioned, OwnedBy, PartOf, ResponseTo) available between these nodes. The relationship between these nodes and edges is sketched below.

User - [CreatesSession] -> TeamChatSession

TeamChatSession - [OwnedBy] -> Team

User - [Joins] -> TeamChatSession

User - [Leaves] -> TeamChatSession

User - [CreateChat] -> ChatItem

ChatItem - [PartOf] -> TeamChatSession

ChatItem - [Mentioned] -> User

ChatItem -> [ResponseTo] -> ChatItem

Creation of the Graph Database for Chats

There are 6 CSV files which contain the relations between nodes and edges. Files are storing these relationships on their rows, and each column corresponds to a node or edge property. Here is the schema of each file's content.

"Chat_create_team_chat.csv" (Columns):

User(Node): This node has a single property UserId indicates each user uniquely

Team(Node): This Node also has a single property TeamId indicates each team

TeamChatSession(Node): This node has a single property SessionId of each team chat

CreatesSession(Edge): This edge indicates the user that created the team chat session

OwnedBy(Edge): This edge relates the chat sessions to teams

“Chat_join_team_chat.csv” (Columns):

User(Node): This node has a single property UserId indicates each user uniquely

TeamChatSession(Node): This node has a single property SessionId of each team chat

Joins(Edge): This edge indicates timestamp that the user joined a team chat session

“Chat_leave_team_chat.csv” (Columns):

User(Node): This node has a single property UserId indicates each user uniquely

TeamChatSession(Node): This node has a single property SessionId of each team chat

Leaves(Edge): This edge indicates timestamp that the user Left a team chat session

“Chat_item_team_chat.csv” (Columns):

User(Node): This node has a single property UserId indicates each user uniquely

TeamChatSession(Node): This node has a single property SessionId of each team chat

ChatItem(Node): This node is the Id of the chat that has transferred between users

CreateChat(Edge): This Edge relates users to their chats that they’ve created

PartOf(Edge): This edge shows the chat item relates to which team chat session

“Chat_mention_team_chat.csv” (Columns):

User(Node): This node has a single property UserId indicates each user uniquely

ChatItem(Node): This node is the Id of chat that has transferred between users

Mentioned(Edge): This edge indicates which users have been mentioned in the chat Items

“Chat_respond_team_chat.csv” (Columns):

ChatItem(Node): This node is the Id of chat that has transferred between users

ChatItem(Node): This node is the Id of chat that has transferred between users

ResponseTo(Edge): This edge indicates which users have been mentioned in the chat Items

To load this data into Neo4j, we should write the schema in CYPHER Code. As an instance, for the second file, I have written below code.

```
LOAD CSV FROM "file:///path/chat_join_team_chat.csv" AS row
```

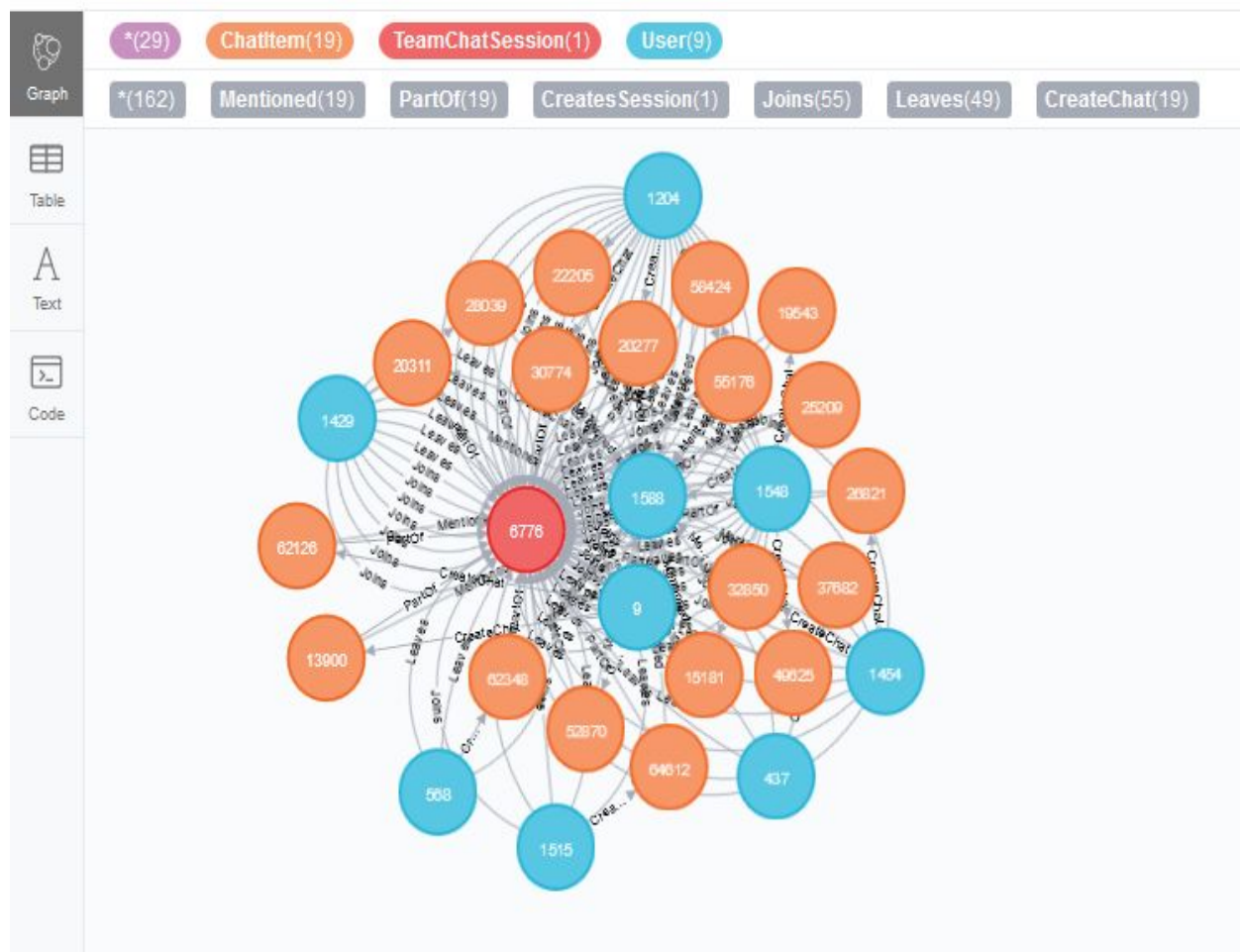
```
MERGE (u:User {id: toInteger(row[0])})
```

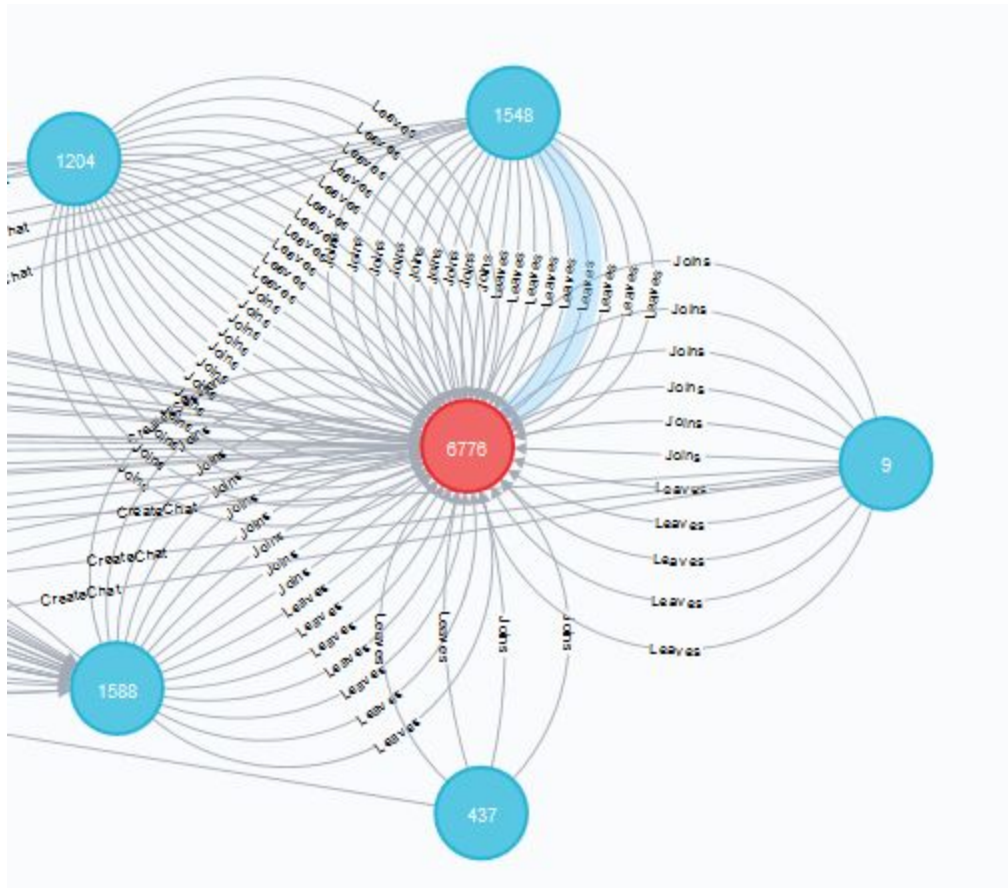
```
MERGE (c:TeamChatSession {id: toInteger(row[1])})
```

```
MERGE (u)-[:Joins{timeStamp: row[2]}]->(c)
```

The code first imports the CSV file as the name row. The second line is defining the first column of the file (row[0]) as the user Id. The third line imports the second column labeled as TeamChatSession with a single property of Id. finally the last line is the edge of this file that relates users to their TeamChatsSessions. It is labeled as joins with a property of timestamp.

```
$ match (n)-[]->( ) return n limit 50
```





Finding the longest conversation chain and its participants

The maximum path length is **9** according to the query below.

```
match p=(a:ChatItem)-[:ResponseTo*]->(c:ChatItem)
```

```
return length(p) order by length(p) desc limit 1
```

This query orders the lengths of responses and shows the first one that is the maximum length available.

The unique users in the longest path of conversation is **5**

Below code shows how the CYPHER code for querying this part of graph.

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

```
with p
```

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

```
where i in nodes(p)
```

```
return count(distinct u)
```

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.

```
match (a)-[:CreateChat*]->(i)
```

```
return distinct count(a.id), a.id
```

```
order by count(a.id) desc limit 10
```

Chattiest Users

Users	Number of Chats
394	115
2067	111
209	109

```
match (i:ChatItem)-[:PartOf]->(c:TeamChatSession)-[:OwnedBy]->(t)
```

```
return distinct count(t.id), t.id
```

```
order by count(t.id) desc limit 10
```

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. According to the first two tables, we get the third table which identifies among the top 10 chattiest users, one of them belongs to chattiest teams, i.e. the user “999” belongs to the team “52”, but other 9 users are not part of the top 10 chattiest teams. This states that most of the chattiest users are not in the chattiest teams. However, we can conclude that there is a relation between the chattiest user and chattiest team and a good strategy is to put some in chat purchasing materials in their chats to make money from that opportunity.

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.

In this question, we will compute an estimate of how “dense” the neighborhood of a node is. In the context of chat that translates to how mutually interactive a certain group of users are. If we can identify these highly interactive neighborhoods, we can potentially target some members of the neighborhood for direct advertising. We will do this in a series of steps.

Most Active Users (based on Cluster Coefficients)

User ID	Coefficient
209	0.93
516	0.90
999	0.78

Recommended Actions

- Putting some effort on Native iPhone apps and make the app faster and more fancy to make the users attracted to spend more time in it. Spending more time in the app can lead to engagement and hence more purchasing.
- Starting some sales funnels specifically for iPhone users to elevate the number of them. This would lead to more HighRoller users and more revenue for the company.
- Based on the clustering analysis, some recommendations have given for each cluster
- Target the chattiest groups and users for making your app atmosphere more competing
- Chattiest users, initiators of longer conversations and users who belong to Chattiest teams and active user groups are probably to be more valuable, because of their potential to spread information to wider audiences. As a result, Eglence, Inc. can increase its revenue by choosing the right marketing strategy to target such users, for example, showing the more expensive items to such users. Even if these users are not going to buy these items, they may influence others in their networks to buy such items.