Data 514: Database Management for Data Science
Final Project

Amazon Sagemaker

Prachi Bambarkar
Jasmine Bhalla
Isaac Kim
Reeya Pimple

I

Introduction

Motivation and Goal for the Project

- As a customer centric eCommerce company with different categories of products, it is essential that Amazon understands how their customers think and make decisions. For this reason, understanding its products and their related data like ratings, co-purchased products is vital so as to make informed decisions for strategy.

- Through this project, we aim to analyze the Amazon dataset by Exploratory Visual Analytics and Predictive Analytics, using Sagemaker to perform the analysis and AWS for data storage.

Tools/System being used:

- **We are primarily using Sagemaker and few other AWS components:**
  - Sagemaker is a fully managed service developed by Amazon Web Services that enables data scientists and developers to quickly and easily build machine-learning models into production smart applications.
    - Instances – We are using Jupyter notebooks for running our queries
  - S3 Bucket to store our data sets
    - We created the bucket **project-sagemaker-3**
  - Other capabilities that can be used for the extension of the project in future:
    - RStudio, Built-in algorithms

II Data Source

We refer to two main datasets for the purpose of this project:

- Amazon product co-purchasing network metadata
- Amazon product co-purchasing network

The data was collected by Jure Leskovec, from Stanford University in 2006, by crawling the Amazon website. It contains information about 548,552 different products

III

Project Workflow

Sagemaker Project Implementation

Research & Planning

Model Development

Testing & Evaluation

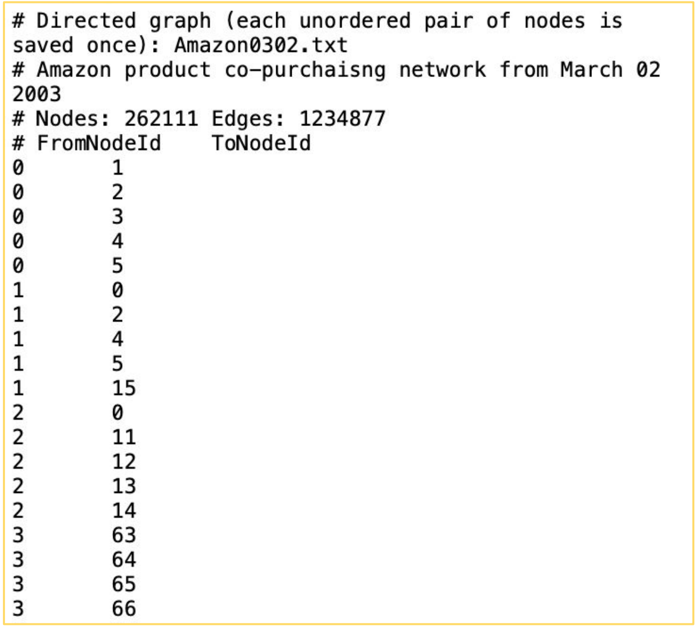
Deployment

IV

Exploratory Data Analysis

Nodes Dataset

Raw Data

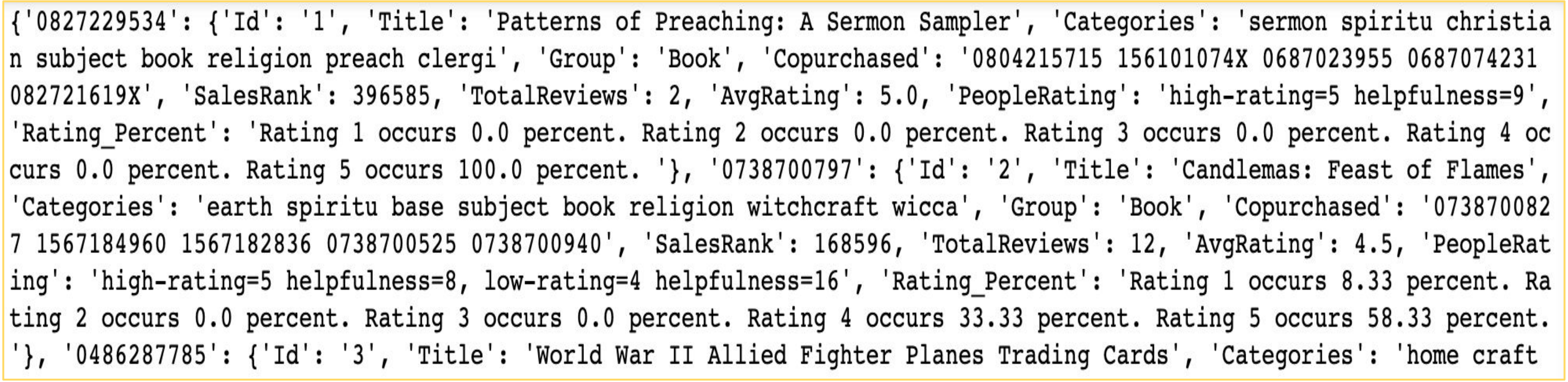


```
# Directed graph (each unordered pair of nodes is
saved once): Amazon0302.txt
# Amazon product co-purchaisng network from March 02
2003
# Nodes: 262111 Edges: 1234877
# FromNodeId    ToNodeId
0       1
0       2
0       3
0       4
0       5
1       0
1       2
1       4
1       5
1       15
2       0
2       11
2       12
2       13
2       14
3       63
3       64
3       65
3       66
```

Data Statistics	
Nodes	262,111
Edges	1,234,877
Diameter (Longest Shortest Path)	32

Products Meta Dataset

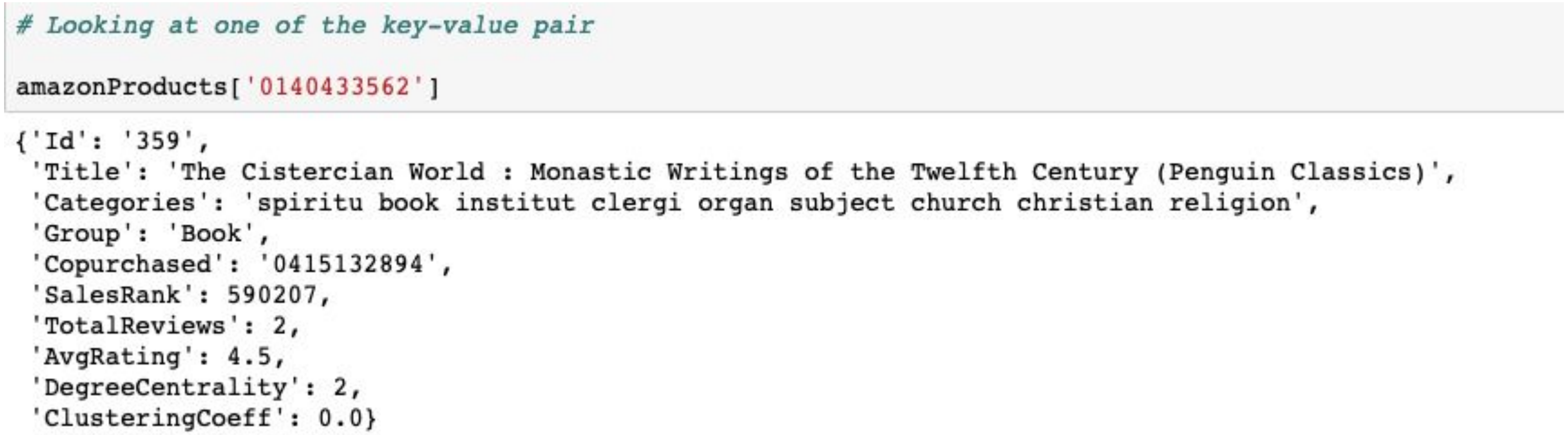
- Used the Amazon meta-data subset to create a nested dictionary- **amazonProducts{}**
  - **Key**- ASIN
  - **Value**- Metadata related to that ASIN which is product ID, Title, Categories, Group, Co-purchased(similar in the metadata file), Sales Rank, Total Reviews, Average Rating, People Rating (highest & lowest rating with helpfulness index) and Rating Percent (The percentage of each rating - between 1 and 5 - given to a product)



```
{'0827229534': {'Id': '1', 'Title': 'Patterns of Preaching: A Sermon Sampler', 'Categories': 'sermon spiritu christia
n subject book religion preach clergi', 'Group': 'Book', 'Copurchased': '0804215715 156101074X 0687023955 0687074231
082721619X', 'SalesRank': 396585, 'TotalReviews': 2, 'AvgRating': 5.0, 'PeopleRating': 'high-rating=5 helpfulness=9',
'Rating_Percent': 'Rating 1 occurs 0.0 percent. Rating 2 occurs 0.0 percent. Rating 3 occurs 0.0 percent. Rating 4 oc
curs 0.0 percent. Rating 5 occurs 100.0 percent. '}, '0738700797': {'Id': '2', 'Title': 'Candlemas: Feast of Flames',
'Categories': 'earth spiritu base subject book religion witchcraft wicca', 'Group': 'Book', 'Copurchased': '073870082
7 1567184960 1567182836 0738700525 0738700940', 'SalesRank': 168596, 'TotalReviews': 12, 'AvgRating': 4.5, 'PeopleRat
ing': 'high-rating=5 helpfulness=8, low-rating=4 helpfulness=16', 'Rating_Percent': 'Rating 1 occurs 8.33 percent. Ra
ting 2 occurs 0.0 percent. Rating 3 occurs 0.0 percent. Rating 4 occurs 33.33 percent. Rating 5 occurs 58.33 percent.
'}, '0486287785': {'Id': '3', 'Title': 'World War II Allied Fighter Planes Trading Cards', 'Categories': 'home craft
```

(The nested dictionary created)

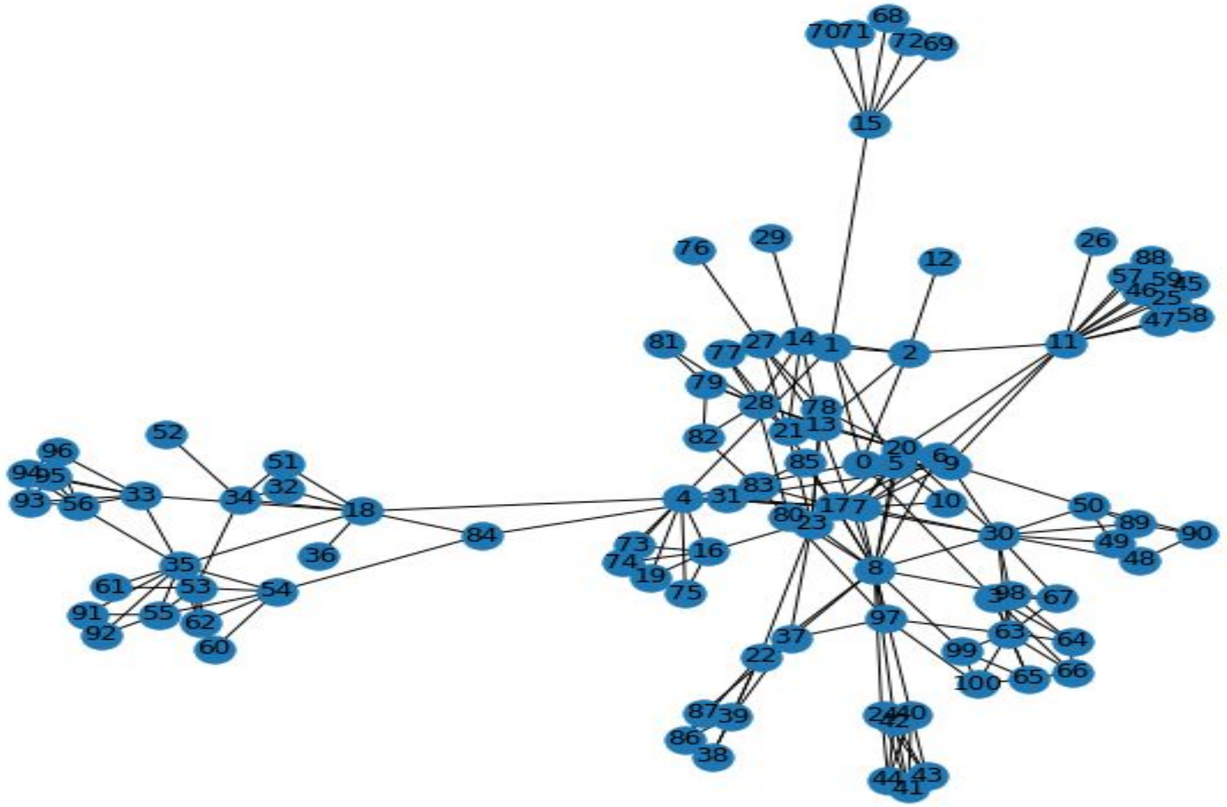
Example of a key-value pair in amazonProducts{}



```
# Looking at one of the key-value pair

amazonProducts['0140433562']
```

```
{'Id': '359',
 'Title': 'The Cistercian World : Monastic Writings of the Twelfth Century (Penguin Classics)',
 'Categories': 'spiritu book institut clergi organ subject church christian religion',
 'Group': 'Book',
 'Copurchased': '0415132894',
 'SalesRank': 590207,
 'TotalReviews': 2,
 'AvgRating': 4.5,
 'DegreeCentrality': 2,
 'ClusteringCoeff': 0.0}
```

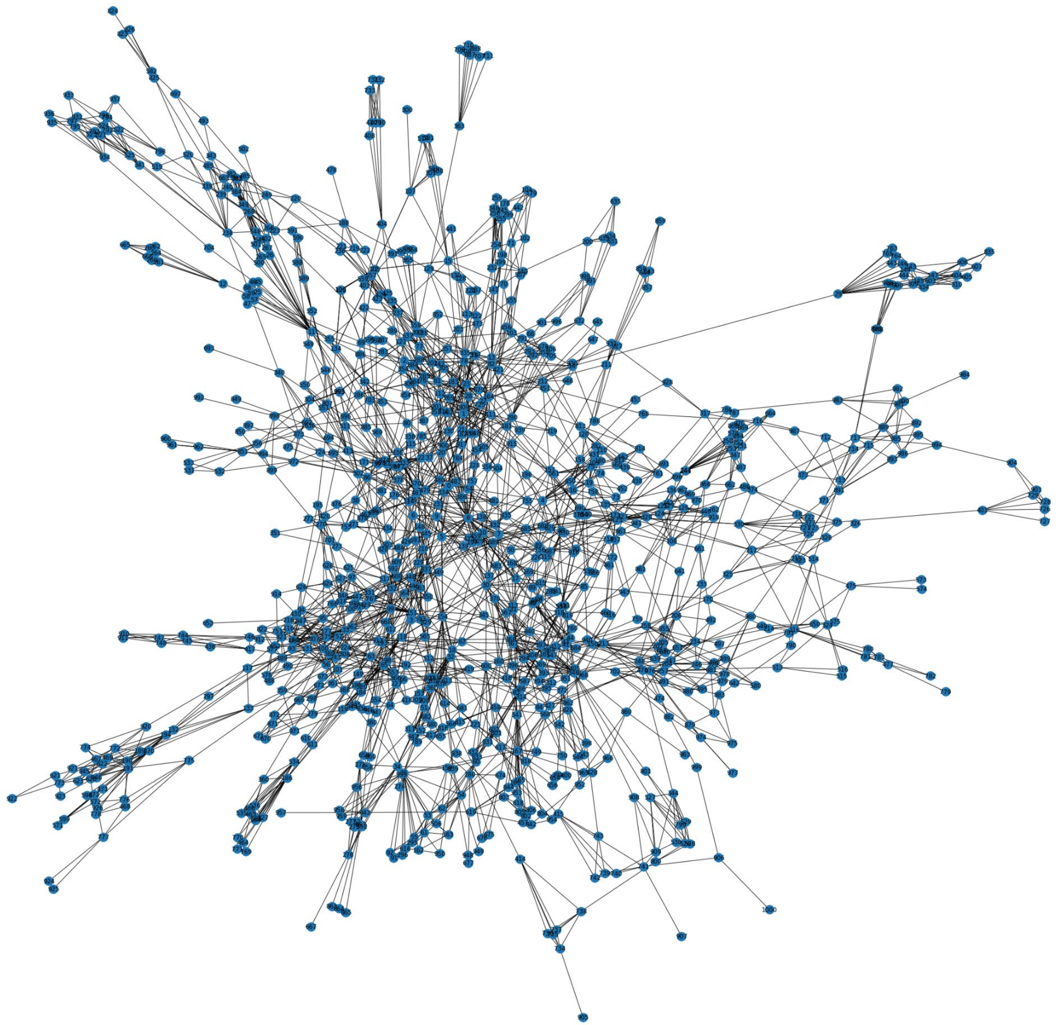
Visualizing 50 Nodes as a Graph



Each Node represents an Amazon Product from the dataset

An Edge from Node A to Node B represents that Customers who purchased B also purchased A.

- **Visualizing 1000 Nodes as a Graph**



V

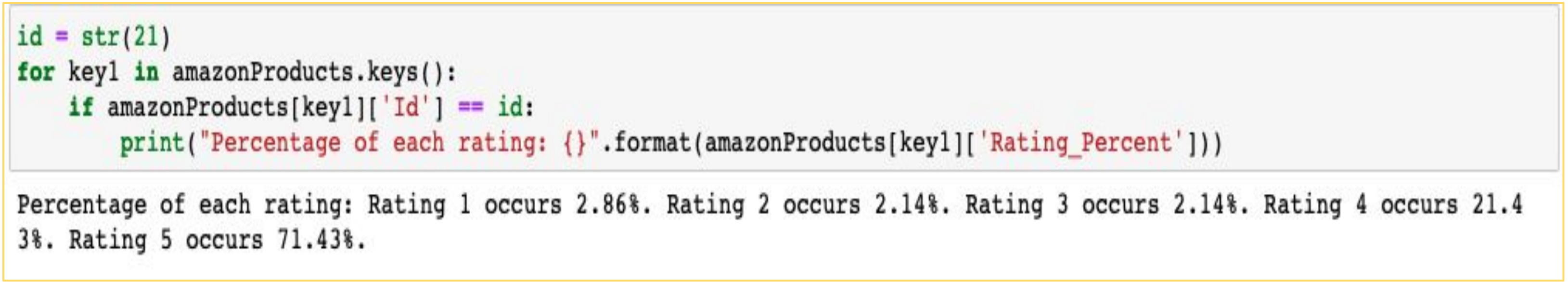
Questions Explored

Q1

Ratings and Variation

What are the percentages of each rating digit for the product with id: 21?

- For this question, we are finding the percentage of each rating (1 - 5) that are available in the ratings given for product with ID: 21.
- This becomes easier since we have created the nested dictionary and added the
- When writing the code for the dictionary, code was added to count the number of each rating in each review available and added to a list. Percentages were calculated from that list.



```
id = str(21)
for key1 in amazonProducts.keys():
    if amazonProducts[key1]['Id'] == id:
        print("Percentage of each rating: {}".format(amazonProducts[key1]['Rating_Percent']))
```

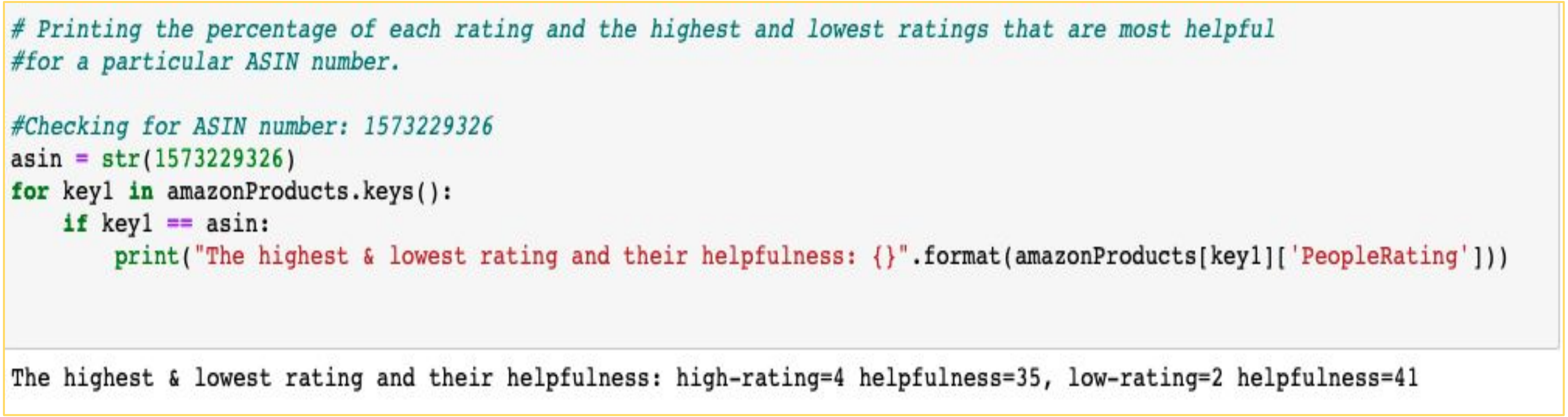
```
Percentage of each rating: Rating 1 occurs 2.86%. Rating 2 occurs 2.14%. Rating 3 occurs 2.14%. Rating 4 occurs 21.4
3%. Rating 5 occurs 71.43%.
```

Q2

Ratings and Variation

For products with ASIN: 1573229326 display the most helpful review(s) of the highest rating and the most helpful review(s) of the lowest rating

- For this question, we are finding the highest rating and lowest ratings that consumers found most helpful.
- While parsing the data, code was added to find the highest and lowest rating available in the reviews given.
- If there was the same rating, then the most helpful rating was displayed.



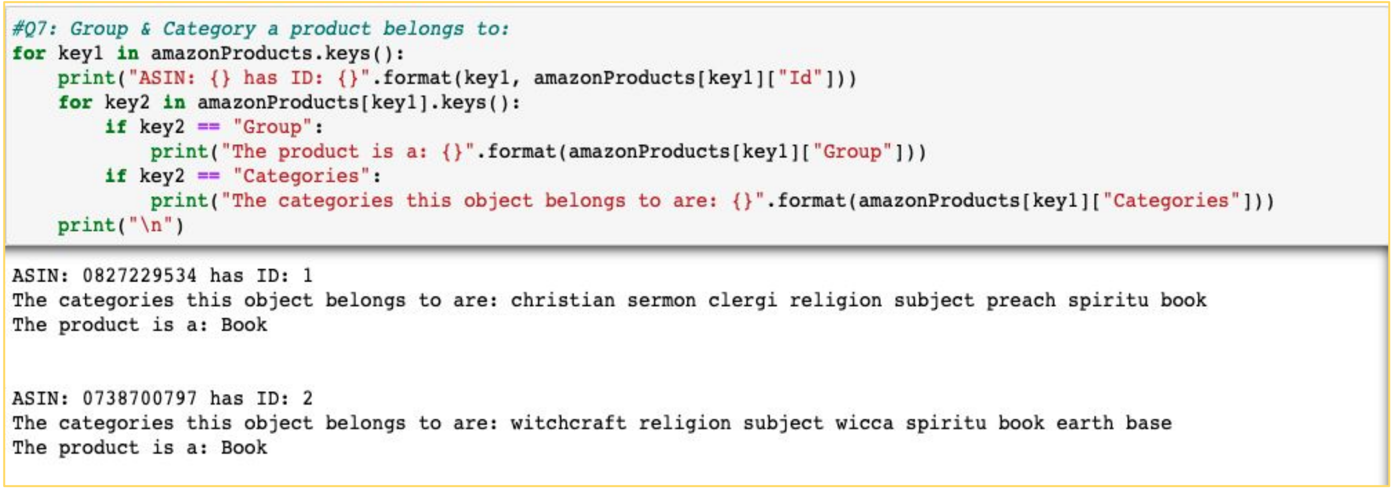
```
# Printing the percentage of each rating and the highest and lowest ratings that are most helpful
#for a particular ASIN number.

#Checking for ASIN number: 1573229326
asin = str(1573229326)
for key1 in amazonProducts.keys():
    if key1 == asin:
        print("The highest & lowest rating and their helpfulness: {}".format(amazonProducts[key1]['PeopleRating']))
```

```
The highest & lowest rating and their helpfulness: high-rating=4 helpfulness=35, low-rating=2 helpfulness=41
```

Q3 Iterating through the nested dictionary

IDs, ASINs, Group and Categories for all products available in the subset

- For this question, iterated through the dictionary to find the IDs, ASINs, Group and Categories that each product belonged to.
- A nested for-loop was used for iteration and we added a print statement to display the desired values.



```
#Q7: Group & Category a product belongs to:
for key1 in amazonProducts.keys():
    print("ASIN: {} has ID: {}".format(key1, amazonProducts[key1]["Id"]))
    for key2 in amazonProducts[key1].keys():
        if key2 == "Group":
            print("The product is a: {}".format(amazonProducts[key1]["Group"]))
        if key2 == "Categories":
            print("The categories this object belongs to are: {}".format(amazonProducts[key1]["Categories"]))
    print("\n")
```

```
ASIN: 0827229534 has ID: 1
The categories this object belongs to are: christian sermon clergi religion subject preach spiritu book
The product is a: Book


ASIN: 0738700797 has ID: 2
The categories this object belongs to are: witchcraft religion subject wicca spiritu book earth base
The product is a: Book
```

Q4

Recommender System

Books recommendation based on **co-purchased data**

Case Study - Books Recommender System

Features:

- amazonBooks{} based on the category- **'Books'** in the amazonProducts{} dictionary
  - Contains total of 97378 books
  - New Graph Metrics associated with Node (ASIN) added:
    - Degree Centrality
    - Clustering Coefficient

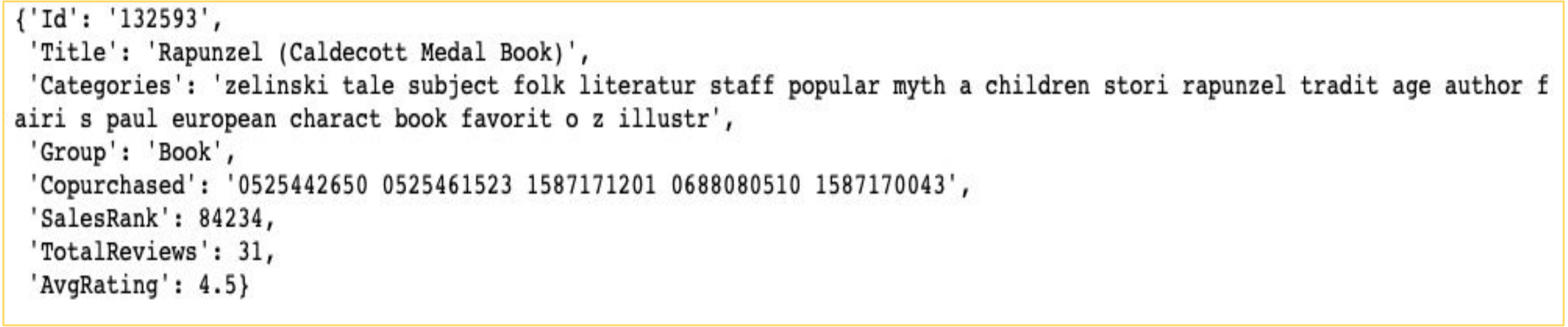
- Takes a ASIN of the book as an input from the user and recommends **Top 5 books** by using the ASINs in CoPurchase/similar metric

- Also, gives out a NodeGraph for that specific ASIN

● Recommender System

Example of key-value pair in amazonBooks{}:

Key: ASIN for a product belonging to the category ‘Book’; **Value:** Metadata related to the ASIN

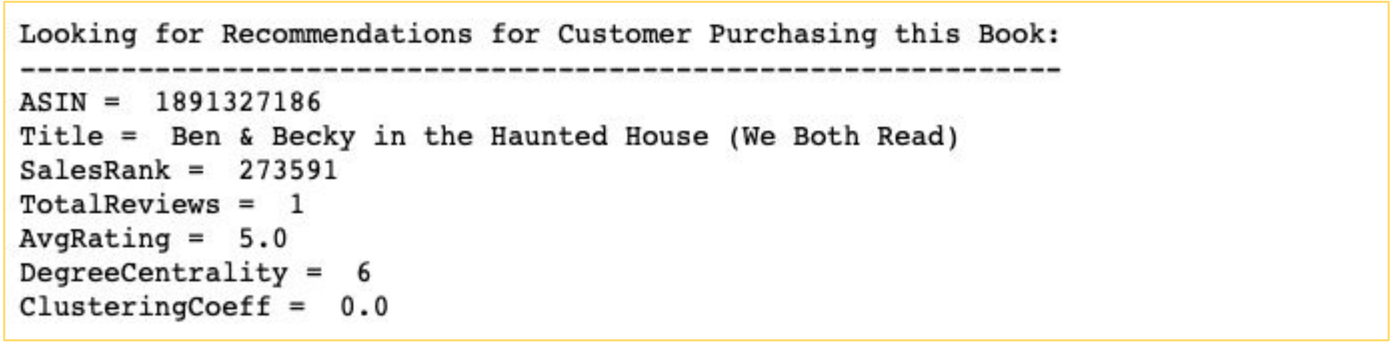


```
{'Id': '132593',
 'Title': 'Rapunzel (Caldecott Medal Book)',
 'Categories': 'zelinski tale subject folk literatur staff popular myth a children stori rapunzel tradit age author f
airi s paul european charact book favorit o z illustr',
 'Group': 'Book',
 'Copurchased': '0525442650 0525461523 1587171201 0688080510 1587170043',
 'SalesRank': 84234,
 'TotalReviews': 31,
 'AvgRating': 4.5}
```

- ## A Quick Run Through

A customer looking for book recommendations for ASIN-1891327186

Metadata information given from amazonBooks{} related to ASIN of the book:

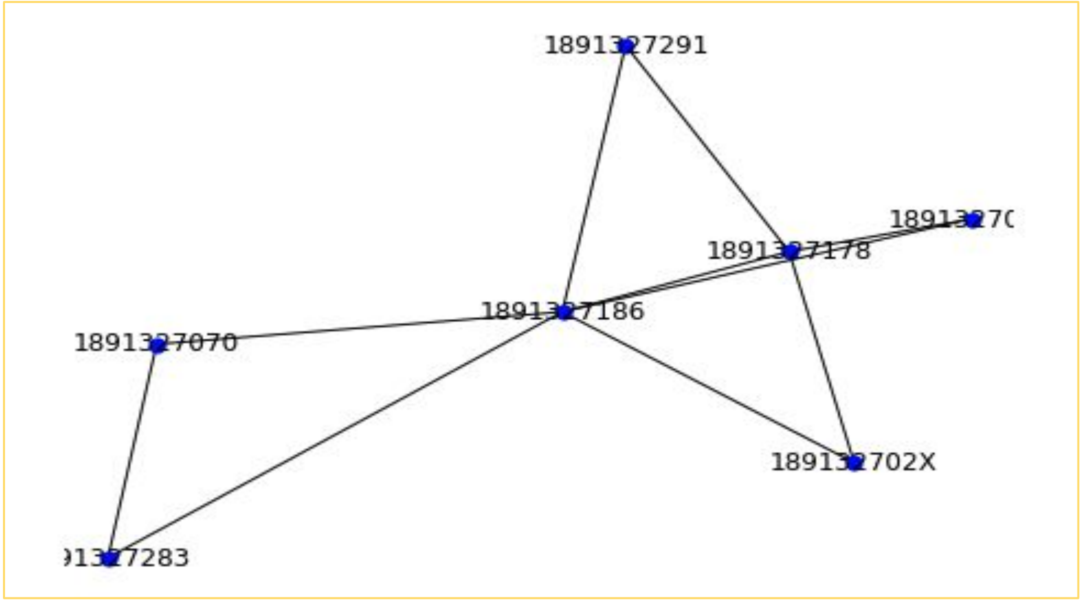


```
Looking for Recommendations for Customer Purchasing this Book:
--------------------------------------------------------------
ASIN =  1891327186
Title =  Ben & Becky in the Haunted House (We Both Read)
SalesRank =  273591
TotalReviews =  1
AvgRating =  5.0
DegreeCentrality =  6
ClusteringCoeff =  0.0
```

● Recommender System

Product Copurchase graph generated for the ASIN

- Graph Nodes - ASIN
- Edges - represent the books that were copurchased
- Edge Weight- represents the similarity between the recommended ASINs

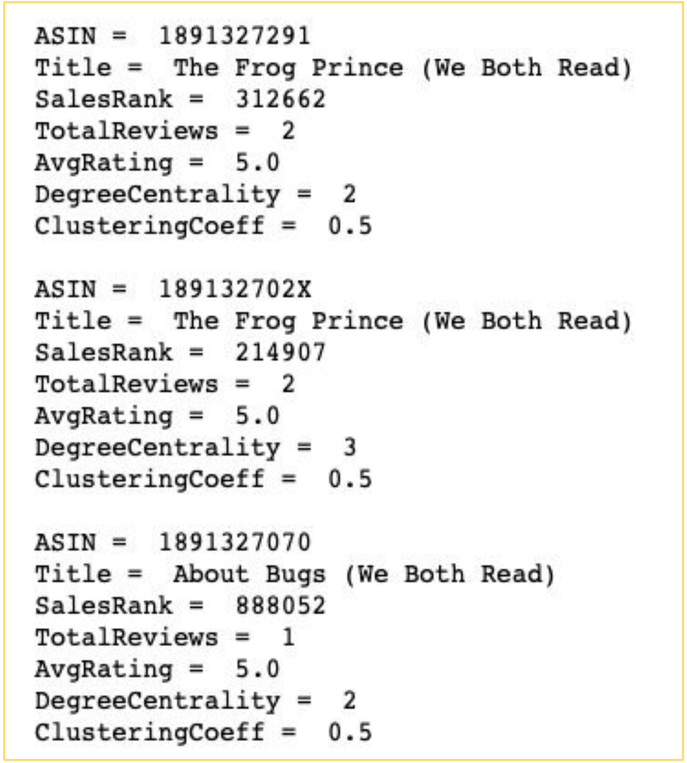


Product Co-Purchase Node Graph for ASIN- **1891327186**

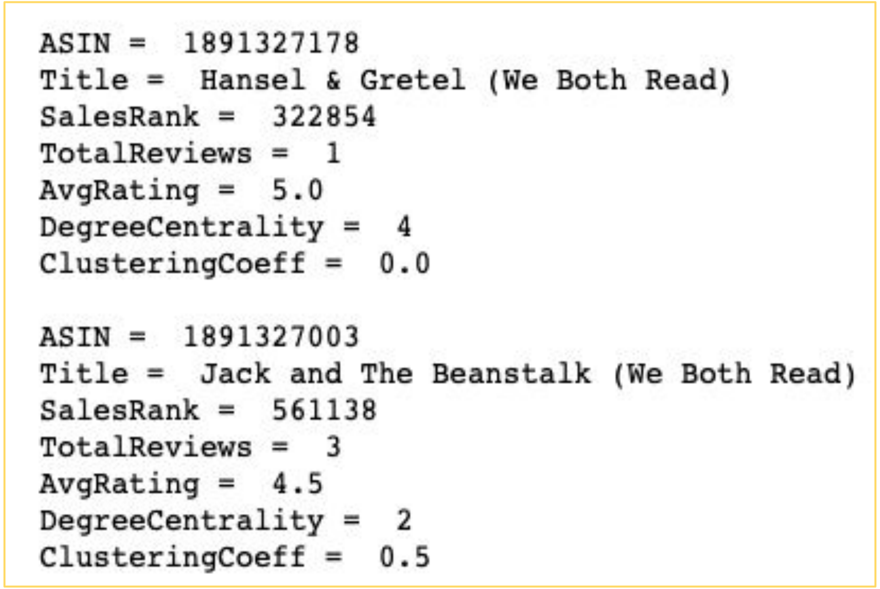
- Number of Nodes: 7
- Number of Edges: 10

● A Quick Run Through

5 Book Recommendations given for ASIN **‘1891327186’**



```
ASIN =  1891327291
Title =  The Frog Prince (We Both Read)
SalesRank =  312662
TotalReviews =  2
AvgRating =  5.0
DegreeCentrality =  2
ClusteringCoeff =  0.5

ASIN =  189132702X
Title =  The Frog Prince (We Both Read)
SalesRank =  214907
TotalReviews =  2
AvgRating =  5.0
DegreeCentrality =  3
ClusteringCoeff =  0.5

ASIN =  1891327070
Title =  About Bugs (We Both Read)
SalesRank =  888052
TotalReviews =  1
AvgRating =  5.0
DegreeCentrality =  2
ClusteringCoeff =  0.5
```



```
ASIN =  1891327178
Title =  Hansel & Gretel (We Both Read)
SalesRank =  322854
TotalReviews =  1
AvgRating =  5.0
DegreeCentrality =  4
ClusteringCoeff =  0.0

ASIN =  1891327003
Title =  Jack and The Beanstalk (We Both Read)
SalesRank =  561138
TotalReviews =  3
AvgRating =  4.5
DegreeCentrality =  2
ClusteringCoeff =  0.5
```

Q5 Pairing Products

Listing pairs of products in the months of March, May, and June of 2003.
Answering question 5

Focus Area

- Behavioral Analysis using “Customers who bought this item also bought”.

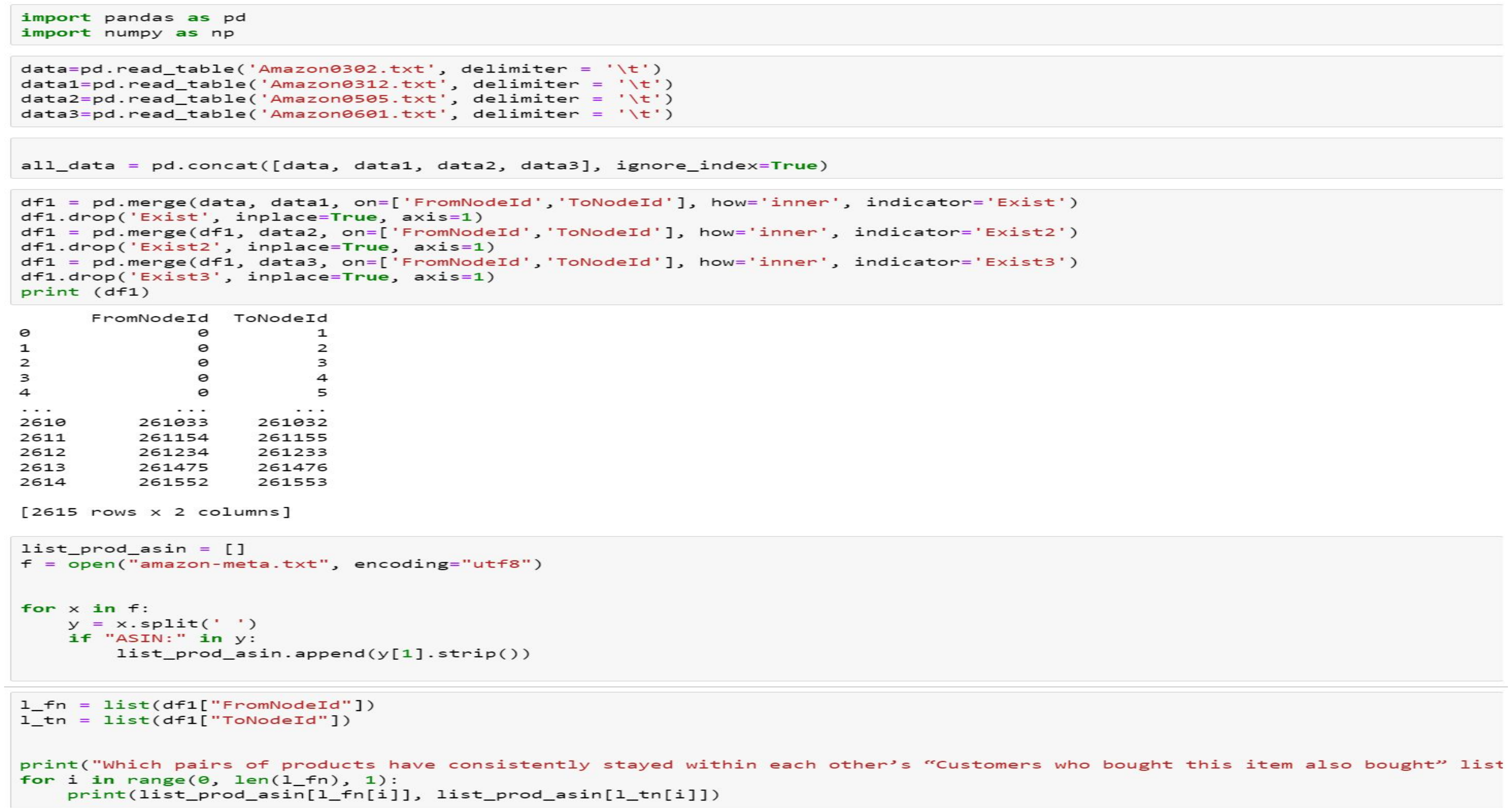
Objective

- List pairs of products pairs of products have consistently stayed within each other’s “Customers who bought this item also bought” lists through march, may and June of 2003.

Dataset in S3 used to answer this question

- Amazon0302, Amazon0312, Amazon0505, Amazon0601, Amazon Metadata

● SageMaker Notebook Instance Runthrough



```
import pandas as pd
import numpy as np
```

```
data=pd.read_table('Amazon0302.txt', delimiter = '\t')
data1=pd.read_table('Amazon0312.txt', delimiter = '\t')
data2=pd.read_table('Amazon0505.txt', delimiter = '\t')
data3=pd.read_table('Amazon0601.txt', delimiter = '\t')
```

```

all_data = pd.concat([data, data1, data2, data3], ignore_index=True)
```

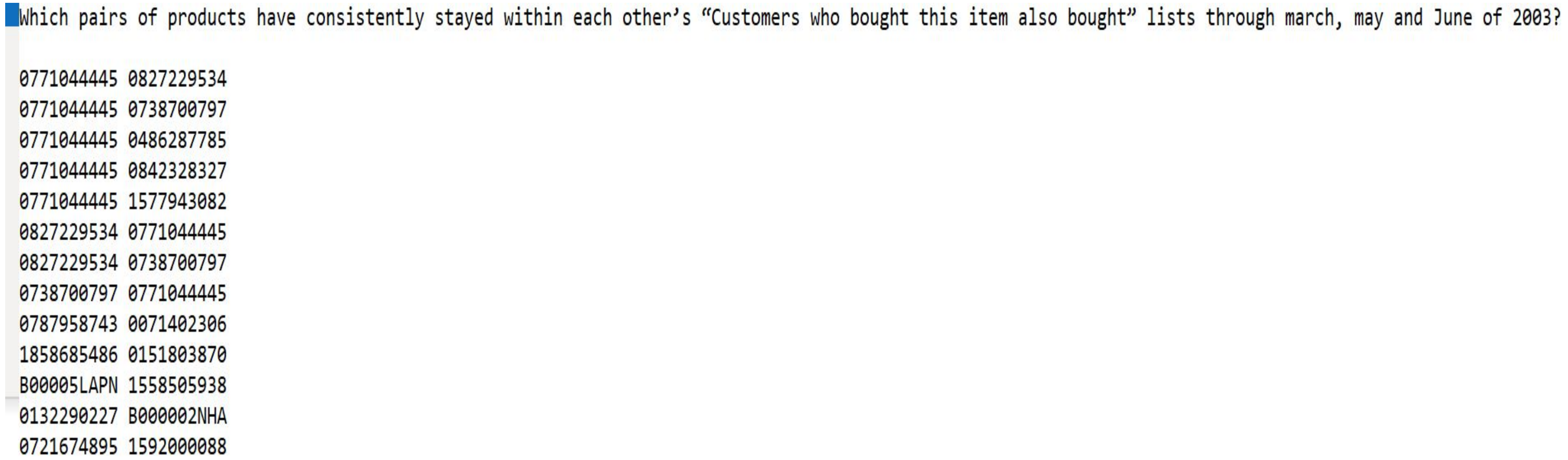
```
df1 = pd.merge(data, data1, on=['FromNodeId','ToNodeId'], how='inner', indicator='Exist')
df1.drop('Exist', inplace=True, axis=1)
df1 = pd.merge(df1, data2, on=['FromNodeId','ToNodeId'], how='inner', indicator='Exist2')
df1.drop('Exist2', inplace=True, axis=1)
df1 = pd.merge(df1, data3, on=['FromNodeId','ToNodeId'], how='inner', indicator='Exist3')
df1.drop('Exist3', inplace=True, axis=1)
print (df1)
```

```
      FromNodeId  ToNodeId
0              0         1
1              0         2
2              0         3
3              0         4
4              0         5
...          ...       ...
2610      261033    261032
2611      261154    261155
2612      261234    261233
2613      261475    261476
2614      261552    261553

[2615 rows x 2 columns]
```

```
list_prod_asin = []
f = open("amazon-meta.txt", encoding="utf8")


for x in f:
    y = x.split(' ')
    if "ASIN:" in y:
        list_prod_asin.append(y[1].strip())
```

```
l_fn = list(df1["FromNodeId"])
l_tn = list(df1["ToNodeId"])


print("Which pairs of products have consistently stayed within each other's "Customers who bought this item also bought" list
for i in range(0, len(l_fn), 1):
    print(list_prod_asin[l_fn[i]], list_prod_asin[l_tn[i]])
```

● Results



```
Which pairs of products have consistently stayed within each other’s “Customers who bought this item also bought” lists through march, may and June of 2003?

0771044445 0827229534
0771044445 0738700797
0771044445 0486287785
0771044445 0842328327
0771044445 1577943082
0827229534 0771044445
0827229534 0738700797
0738700797 0771044445
0787958743 0071402306
1858685486 0151803870
B00005LAPN 1558505938
0132290227 B000002NHA
0721674895 1592000088
```

Q6

Finding Shortest Path between Nodes

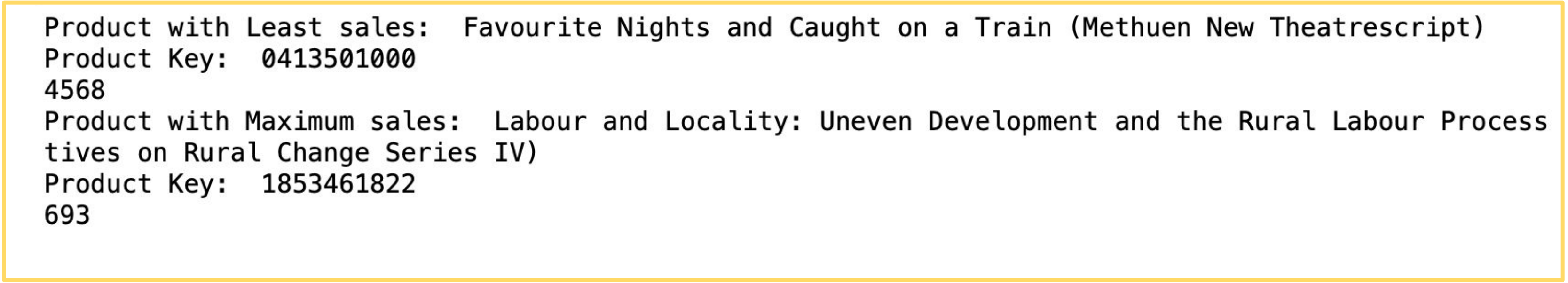
If a user is at the amazon.com page for the bestselling product and from then on is only using the links under “Customers who bought this item also bought” list to view other items, how many items (clicks) later will they reach the worst selling product? What was the path taken? List the items in the order in which they were clicked.

Steps:

- The amazon0302.txt contains Edges data for all IDs present in metadata
- Dataset is parsed to create a list of Edges
  - Store Product with Highest Sales and Lowest Sales
- To find the Shortest Path:
  - Create a adjacency list from graph
  - Using **Breadth First Search** (BFS) find shortest path between two selected nodes
- Output the resultant path with titles of each product

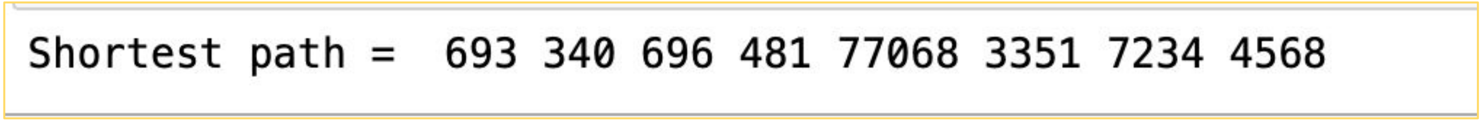
Finding Shortest Path between Nodes

Identifying the Highest Selling and Lowest Selling Product



```
Product with Least sales:  Favourite Nights and Caught on a Train (Methuen New Theatrescript)
Product Key:  0413501000
4568
Product with Maximum sales:  Labour and Locality: Uneven Development and the Rural Labour Process
tives on Rural Change Series IV)
Product Key:  1853461822
693
```

Finding the Shortest Path between the two nodes



```
Shortest path =  693 340 696 481 77068 3351 7234 4568
```

Q7

Identifying Similar Items

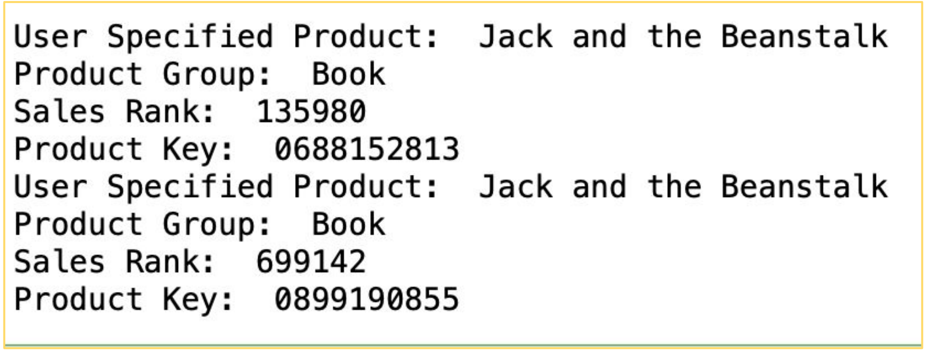
For any user specified product A (“Jack and the Beanstalk”) belonging to group G, which products are in its CoPurchase list? And which groups do those products belong to?

Steps:

- Identify all products with the title “Jack and the Beanstalk”
- Since we obtain multiple products, select the product with the highest sales
- Print all Products matching the ‘Id’ number in the Co-purchase list

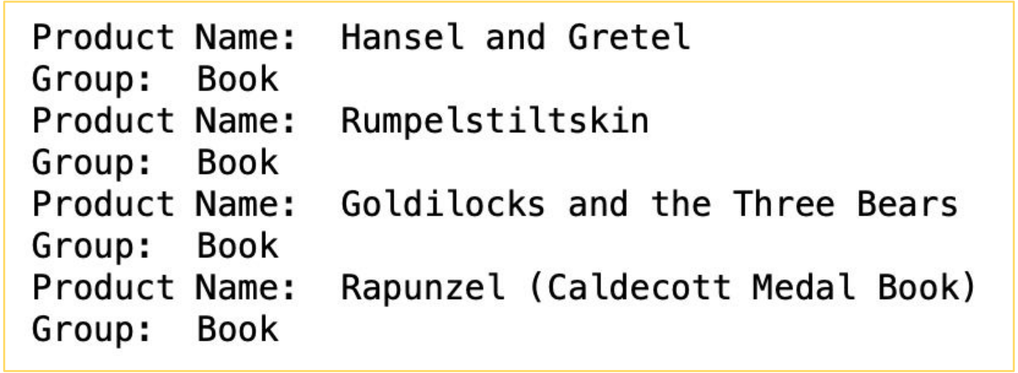
● Identifying Similar Items

Printing all items that match “Jack and the Beanstalk”



```
User Specified Product:  Jack and the Beanstalk
Product Group:  Book
Sales Rank:  135980
Product Key:  0688152813
User Specified Product:  Jack and the Beanstalk
Product Group:  Book
Sales Rank:  699142
Product Key:  0899190855
```

Printing all items in the CoPurchase list



```
Product Name:  Hansel and Gretel
Group:  Book
Product Name:  Rumpelstiltskin
Group:  Book
Product Name:  Goldilocks and the Three Bears
Group:  Book
Product Name:  Rapunzel (Caldecott Medal Book)
Group:  Book
```

VI

Lessons Learned

Lessons Learned

- Parsing the data can be a really time consuming and frustrating task
- Exploring a new system can be hard
- Automating everything like creation of S3 buckets, versions of models, etc. is the key
- The use of Jupyter notebooks in Sagemaker ensures that the users can use it easily

VII

Future Scope

- Build segments and predictions using SageMaker.
- Use propensity scores for display targeting.
- Send personalized messaging using Amazon Pinpoint.
- Integrate real-time personalized suggestions using Amazon Personalize.

VIII Sources

Datasets:

- http://snap.stanford.edu/data/amazon0302.html
- https://snap.stanford.edu/data/amazon-meta.html

References:

- https://www.geeksforgeeks.org

Thankyou