Final Project

Olivia(Peiyu) Wang
Madison Turano

Final Project Components

1. Sentiment Analysis
2. User Analysis
3. Comparison of MongoDB and MySQL

Dataset Introduction & Database Setup

- Amazon Reviews: Kindle Store Category
- Features: asin, helpful, overall, reviewText, reviewTime, reviewerID, reviewerName, summary, UnixReviewTime
- Dropped helpful because we cannot interpret it properly and reviewTime because it’s not meaningful to our analysis

Database Setup

- We used the json file to set up both databases
- For MongoDB we directly imported the whole json file into MongoDB
- For MySQL we had two tables set up- Reviewer and Review.
  - Review table is connected to Reviewer table with ReviewerID as the foreign key

Database - MongoDB

Key	Value
(1) ObjectId("5cb6729966c8523048f1b08a")	{ 10 fields }
_id	ObjectId("5cb6729966c8523048f1b08a")
reviewerID	A1F6404F1VG29J
asin	B000F83SZQ
reviewerName	Avidreader
helpful	[2 elements]
reviewText	I enjoy vintage books and movies so I enjoyed reading this book. The plot was u...
overall	5.0
summary	Nice vintage story
unixReviewTime	1399248000
reviewTime	05 5, 2014

Database - MySQL, Review

review_id	asin	overall	review_text	reviewer_id	reviewer_name	summary
0	B000F83SZQ	5	I enjoy vintage books and movies so I enjoyed readi..	A1F6404F1VG29J	Avidreader	Nice vintag
1	B000F83SZQ	4	This book is a reissue of an old one; the author wa..	AN0N05A9LIJEQ	critters	Different..
2	B000F83SZQ	4	This was a fairly interesting read. It had old- st..	A795DMNCJILA6	dot	Oldie
3	B000F83SZQ	5	I'd never read any of the Amy Brewster mysteries un..	A1FV0SX13TWVXQ	Elaine H. Turley "Montana Songbird"	I really li
4	B000F83SZQ	4	If you like period pieces - clothing, lingo, you wi..	A3SPTOKDG7WBLN	Father Dowling Fan	Period Myst
5	B000F83SZQ	4	A beautiful in-depth character description makes it..	A1RK2OCZDSGC6R	ubavka seirovska	Review
6	B000F83SZQ	4	I enjoyed this one tho I'm not sure why it's called..	A2HSAKHC3IBRE6	Wolfmist	Nice old fa
7	B000F83SZQ	4	Never heard of Amy Brewster. But I don't need to li..	A3DE6XGZ2EPADS	WPY	Enjoyable r
8	B000FA64PA	5	Darth Maul working under cloak of darkness committi..	A1UG4Q4D3OAH3A	dsa	Darth Maul
9	B000FA64PA	4	This is a short story focused on Darth Maul's role ..	AQZH7YTWQPOBE	Enjolras	Not bad, no
10	B000FA64PA	5	I think I have this one in both book and audio. It ..	A1ZT7WV0ZUA0OJ	Mike	Audio and b
11	B000FA64PA	4	Title has nothing to do with the story. I did enjo..	A2ZFR72PT054YS	monkeyluis	Darth Maul.
12	B000FA64PA	3	Well written. Interesting to see Sideous (through M..	A2QK1U70OJ74P	Sharon Deem	Not bad; it
13	B000FA64PK	3	Troy Denning's novella Recovery was originally publ..	A3SZMGJMV0G16C	Andrew Pruette "Rancors Love to Read"	Han and Lei
14	B000FA64PK	5	I am not for sure on how much of a difference the s..	A3H8PE1UFK04JZ	Caleb Watts	Possibly Im
15	B000FA64PK	5	I really enjoyed the book. Had the normal back agai..	A2EN84QHDRZLP2	Carl craft	Another rea
16	B000FA64PK	5	Great read enjoyed every minute of it . I think it..	A1UG4Q4D3OAH3A	dsa	Recovery
17	B000FA64PK	3	Another well written eBook by Troy Denning, but why..	A38Z3Q6DTDIH9J	Jimmy J. Shaw "oldbent1"	Star Wars:
18	B000FA64PK	5	This one promises to be another good book. I have b..	A1ZT7WV0ZUA0OJ	Mike	my collecti
19	B000FA64PK	4	I have a version of "Star by Star" that does not in..	A22CW0ZHY3NJH8	Noname	Not a neces

Database - MySQL, Reviewer

reviewer_id	reviewer_name
A00085083TSCV82430YT4	Ja'netHayes
A0010876CNE3ILIM9HV0	JassyR
A00207583M69Q8KX3BOFQ	debra wolstenholme
A002359833QJM7OQHCXWY	Dawn Peterson
A00328401T70RFN4P1IT6	Susan
A00463782V7TKAP9EMNL	Diane
A006458827ALF2J23JJTO	Matthew Fudge
A0089401235VSN3Z6F3HK	Draven Valverde
A0090953K7LNUG6UPMI6	Alice Herde
A0092581WFYQNV4KMUZ3	Jody
A0093003C4D9BVJ1YFA	Kindle Customer
A0099735VDZ3HDCAAYKL	M.Asa "MELVENA ASA"
A01024073VQNJIY6SIY50	Christine Zamora
A01097111 3OD625HDB6X8	BookaddictBieke "Istyria book blog"

Sentiment Analysis

Questions/Goals:

- Predict overall rating based on review and summary
- The Comparison of completing the task with MongoDB and AWS MySQL

Outline

1. Loading data from MongoDB
2. Loading data from MySQL
3. Data Preprocessing
4. Machine Learning Algorithm

MongoDB - Access Database

Step 1: Setup Client

```
client = MongoClient(host='localhost',
                     port=27017)
```

Step 2: Access Database

```
db = client.kindle_reviews

#Proof we connected to database
collection = db.reviews
doc = collection.find_one({})
print(doc)
```

{'_id': ObjectId('5cb6729966c8523048f1b08a'), 'reviewerID': 'A1F6404F1VG29J', 'asin': 'B000F83SZQ', 'reviewerName': 'Avidreader', 'helpful': [0, 0], 'reviewText': "I enjoy vintage books and movies so I enjoyed reading this book. The plot was unusual. Don't think killing someone in self-defense but leaving the scene and the body without notifying the police or hitting someone in the jaw to knock them out would wash today.Still it was a good read for me.", 'overall': 5.0, 'summary': 'Nice vintage story', 'unixReviewTime': 1399248000, 'reviewTime': '05 5, 2014'}

MongoDB - Load Data

- For accurate analysis, need balanced dataset
    - I.e. same number of records per rating
- Smallest rating subset is 1 with 23,018 records
- Randomly select 20,000 records per rating subset

MongoDB - Random Selection

```
five_subset = list(collection.aggregate([{'$match': {'overall': 5}},
                                          {'$sample': {'size': 20000}}]))
four_subset = list(collection.aggregate([{'$match': {'overall': 4}},
                                          {'$sample': {'size': 20000}}]))
three_subset = list(collection.aggregate([{'$match': {'overall': 3}},
                                          {'$sample': {'size': 20000}}]))
two_subset = list(collection.aggregate([{'$match': {'overall': 2}},
                                          {'$sample': {'size': 20000}}]))
one_subset = list(collection.aggregate([{'$match': {'overall': 1}},
                                          {'$sample': {'size': 20000}}]))
```

MongoDB - Merge Subsets

```
review_data = pd.DataFrame(columns = ['id', 'reviewText', 'summary', 'overall'])
subsets = [five_subset, four_subset, three_subset, two_subset, one_subset]
```

```
for subset in subsets:
    for j in range(20):
        for i in range(j*1000, (j+1)*1000):
            _id = subset[i]["_id"]
            reviewText = subset[i]['reviewText']
            summary = subset[i]['summary']
            overall = subset[i]['overall']
            review_data = review_data.append({'id': _id,
                                    'reviewText':reviewText,
                                    'summary': summary,
                                    'overall': overall},
                                    ignore_index = True)
```

MongoDB - Dataset

```
print(review_data.shape)
review_data = review_data.sort_values(by=['id'])
review_data.head()
```

```
(100000, 4)
```

	id	reviewText	summary	overall
44109	5cb6729966c8523048f1b096	well written interesting to see sideous throug...	not bad it is well written	3.0
52461	5cb6729966c8523048f1b097	troy dennings novella recovery was originally ...	han and leia reunited and barabel jedi introduced	3.0
44934	5cb6729966c8523048f1b09b	another well written ebook by troy denning but...	star wars the new jedi order recovery	3.0
62888	5cb6729966c8523048f1b09f	with ylesia a novella originally published in ...	minor new jedi order side story	2.0
1379	5cb6729966c8523048f1b0a4	really shouldnt have han solo on the cover as ...	an interesting short story	5.0

MySQL - Randomly Select Data

```
five_subset = pd.read_sql("SELECT overall, review_text, summary FROM review WHERE overall = 5 ORDER BY RAND() LIMIT 2000
four_subset = pd.read_sql("SELECT overall, review_text, summary FROM review WHERE overall = 4 ORDER BY RAND() LIMIT 2000
three_subset = pd.read_sql("SELECT overall, review_text, summary FROM review WHERE overall = 3 ORDER BY RAND() LIMIT 200
two_subset = pd.read_sql("SELECT overall, review_text, summary FROM review WHERE overall = 2 ORDER BY RAND() LIMIT 20000
one_subset = pd.read_sql("SELECT overall, review_text, summary FROM review WHERE overall = 1 ORDER BY RAND() LIMIT 20000
```

MySQL - Randomly Selection Closeup

```
SELECT review_id, overall, review_text, summary FROM review
WHERE overall = 1
ORDER BY RAND()
LIMIT 20000;
```

Data Preprocessing - Remove Common Words

```python
#Remove common words

stop_words = set(stopwords.words('english'))

def text_cleaner(review):
    word_tokens = word_tokenize(review)
    filtered_sentence = [w for w in word_tokens if not w in stop_words]
    filtered_sentence = []
    for w in word_tokens:
        if w not in stop_words:
            filtered_sentence.append(w)

    s = ' '
    clean_sentence = s.join(filtered_sentence)
    return clean_sentence
```

Data Preprocessing - Vectorize Text

```
from sklearn.feature_extraction.text import CountVectorizer

cv = CountVectorizer(binary=True)

cv.fit(review_train['reviewText'])
reviewText_train = cv.transform(review_train['reviewText'])

cv.fit(review_test['reviewText'])
reviewText_test = cv.transform(review_test['reviewText'])

cv.fit(review_train['summary'])
reviewSummary_train = cv.transform(review_train['summary'])

cv.fit(review_train['summary'])
reviewSummary_test = cv.transform(review_test['summary'])
```

- Creates matrix
- Row = sentence, Column = word
- 1 = word in sentence 0 = word not in sentence

Machine Learning - Setup

```
model = torch.nn.Sequential(
    torch.nn.Linear(input_size, hidden_size),
    torch.nn.ReLU(),
    nn.LayerNorm(hidden_size),
    nn.ReLU().cuda(),
    nn.Linear(hidden_size, num_classes))
```

- Pytorch (aka torch): Open source machine learning algorithm
- Use neural network to process data and predict rating
- Like sklearn but more powerful

Machine Learning - Train and Test

```
for epoch in range(num_epochs):
    for i in range(input_size):
        # Convert torch tensor to Variable
        text = reviewText_train.toarray()[i]
        text = torch.from_numpy(text)
        rating = np.array(review_train['overall'])[i]
        rating = torch.from_numpy(rating)

        # Forward + Backward + Optimize
        optimizer.zero_grad()  # zero the gradient buffer
        output = model(text)
        loss = criterion(output, rating)
        loss.backward()
        optimizer.step()
    print(epoch, loss.item())
    epochs = np.append(epochs, index)
    loss_index = np.append(loss_index, loss.item())
```

Key Differences

MongoDB	MySQL
Create collection to call data	Call data with one command line
Load subsets as lists	Load subsets as dataframes
Append data to empty dataframe	Merge dataframes
More lines of code but fast	Lines of code but slow loading

User Analysis

Questions/Goals:

- What products do users rate higher/lower?
- What is range of users' ratings?
- The Comparison of completing the task with MongoDB and AWS MySQL

Sampling

Sampling Procedure:

- There are about 980,000 reviews in this dataset
- 68,000 unique reviewers in the dataset
- In order to finish this task, sampling is necessary

Sampling

- Used MySQL database to query the unique reviewer list since reviewerID is the primary key in the Reviewer table
  - This ensured the uniqueness of reviewerID in Reviewer table which made it easier to query all the unique reviewers.
- Shuffled the whole unique reviewerID list and selected the first 5000 ReviewerID as my sample
- Copied and pasted the unique ReviewerID list to MongoDB analysis to make sure that the reviewers I am analyzing are the same

Querying Data from MySQL

```
def rating(reviewer_id):
    rating_list={}
    asin_list=[]
    res=cursor.execute(f"SELECT asin FROM review WHERE reviewer_id='{reviewer_id}'")
    asin = cursor.fetchall()
    for i in asin:
        rating_list_for_specific_product=[]
        #asin_list.append(i[0])#all the asin of the products that a specific reviewer has reviewed are in the asin_lis
        res_2=cursor.execute(f"SELECT overall FROM review WHERE reviewer_id='{reviewer_id}' AND asin='{i[0]}'")
        ratings=cursor.fetchall()
        for j in ratings:
            rating_list_for_specific_product.append(j[0])
        #print(rating_list_for_specific_product)
        rating_list[i[0]]=rating_list_for_specific_product
    return rating_list
```

Querying Data from MongoDB

```
all_rating_dict={}
for i in only_ID_list:
    product_rating_dict={}
    product_list=[]
    docs=coll.find({"reviewerID":i})
    for doc in docs:
        product_list.append(doc["asin"])
    #print(product_list)
    for product in product_list:
        #print(product)
        rating_list=[]
        docs=coll.find({"$and":[{"reviewerID":i},{"asin":product}]})
        for doc in docs:
            #print(doc["asin"])
            #print(doc["overall"])
            rating_list.append(doc["overall"])
        product_rating_dict[product]=rating_list
        print(product_rating_dict)
    all_rating_dict[i]=product_rating_dict
print(all_rating_dict)
```

Results

{'A31LSUL83B7JA': {'B00C404E60': [5], 'B00FRKKFBW': [5], 'B00GR4XFPU': [4], 'B00HD6IQDM': [5], 'B00HUHUA2O': [5]}, 'A
39F8O421OZHNL': {'B0042RUKGG': [4], 'B005U3GXO4': [5], 'B006IXPHBK': [5], 'B006M3X2XQ': [5], 'B007ED8Y7M': [5], 'B007
IQC6TC': [4], 'B007VWHFNU': [5], 'B0094B0WFY': [5], 'B009JGYOQM': [5], 'B00A8G2CIY': [5], 'B00A8TUS8M': [5], 'B00AXGF
IB2': [4]}, 'A3LIAXZO1OZ6OS': {'B004M8S850': [5], 'B0050JL082': [3], 'B0081VXDZY': [5], 'B008QPW5J0': [5], 'B009XOKN9
2': [5], 'B00A05XOX0': [5], 'B00BWY3UKU': [5], 'B00F1MU458': [5]}, 'A32AG8E8C355JR': {'B005D1L7VM': [5], 'B0079JHOF4'
: [5], 'B007FM513W': [5], 'B007N9B9YC': [5], 'B008OP7WJU': [5], 'B00EODERRG': [5], 'B00FU9RIGK': [5]}, 'A16UICWG95JJY
E': {'B00413QQ1E': [5], 'B005VGNELU': [5], 'B006HVW2H0': [5], 'B00ASQS51M': [5], 'B00GA2X5PO': [5]}, 'A1V8FNQJFK2RVE'
: {'B0055ECOUA': [5], 'B0057QO8Q4': [5], 'B0061YAUG8': [5], 'B0083ZJ86G': [4], 'B00C559VUI': [4]}, 'A1SCBO2BIYL8H2':
{'B008674PGO': [5], 'B009JF5ZY8': [5], 'B00HCFHXZ6': [5], 'B00IDWEJ40': [5], 'B00IMKD22I': [5], 'B00JRBN26M': [4], 'B
00K0KH74C': [4], 'B00K7J786S': [5], 'B00LKSISBO': [5]}, 'A24WKCA4H3A6MA': {'B008R9JFQ6': [5], 'B009Q63QGE': [5], 'B00
B5HJMHO': [5], 'B00B8GHMEC': [5], 'B00DJTK6LM': [5]}, 'A3LUNCIQS0JQW': {'B00BANV988': [5], 'B00C0EJC7Q': [3], 'B00CCR
TFSC': [5], 'B00F9B09WY': [5], 'B00G5XIXDM': [5], 'B00H1G6ZGE': [5], 'B00HBSNH6S': [3], 'B00HXR4Y5U': [4], 'B00IOOZY6
E': [5]}, 'AK6VV0OQJ6XA8': {'B008BJ268Q': [4], 'B009KGAF9Q': [5], 'B00D6IAJHM': [4], 'B00DKDT1PO': [4], 'B00G4QZY78':
[4], 'B00IRJ99B2': [4], 'B00IWQ3M0Y': [5], 'B00J7FLV6Q': [4]}, 'AFXI9B3MIKWM8': {'B005LJX21O': [5], 'B006ITXEVO': [5]
, 'B00757ZY2K': [5], 'B007R5T3SQ': [5], 'B00A3XQW9M': [5], 'B00CFAGTV2': [5], 'B00EHSUFD8': [5], 'B00IFTK8CS': [5]},
'A32YEW2XB0LLBE': {'B0055TFOM0': [5], 'B008E95C20': [5], 'B009YOISDY': [5], 'B00F9AO81S': [5], 'B00GGKHFW4': [5]}, 'A
3BBM4UHPXVJS9': {'B002DOSBO8': [3], 'B0042RUKX4': [1], 'B004PLO6GO': [3], 'B004U362DC': [5], 'B004W3UD6C': [2], 'B005
22PBLY': [1], 'B005A7U328': [1], 'B005OBMIZA': [2], 'B007HOMP78': [2], 'B007XVV6PW': [4], 'B008EN40FG': [2], 'B00ANBR
9FK': [3], 'B00AV36H44': [5], 'B00C1N959Q': [4], 'B00CLHILME': [1], 'B00DOZ8PT6': [1], 'B00DR4Y0AM': [1], 'B00F22NSP0

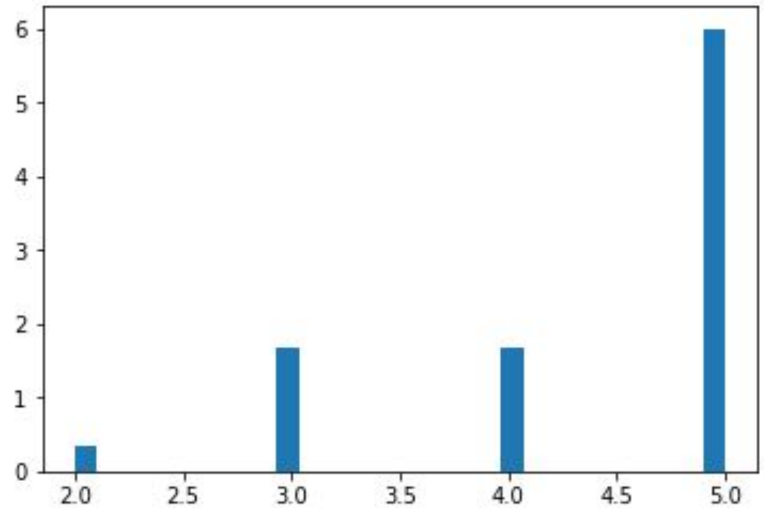
Reviewer Rating Analysis Part.1

```
user_rating=all_user_rating_dict['A31LSUL83B7JA'].values()
print(user_rating)
#or directly query from MySQL database using A00085083TSCV82430YT4_rating_list=rating('A00085083TSCV82430YT4'); print(
import numpy as np
user_rating_list=[]
for i in user_rating:
    print(i)
    user_rating_list.append(i[0])
user_average_rating=np.mean(user_rating_list)
print(user_average_rating)
```

```
dict_values([[5], [5], [4], [5], [5]])
[5]
[5]
[4]
[5]
[5]
4.8
```

User Rating Analysis Part.2

```
4.379310344827586
```




```
[5]
[5]
[5]
[5]
[5]
[4]
[5]
[5]
[2]
[5]
[5]
[4]
[5]
[3]
[5]
[5]
[5]
[4]
[4]
[5]
[5]
[5]
[3]
[3]
[3]
[3]
[4]
[5]
[5]
```

Look into the specifics

```
all_user_rating_dict['A13ELLBM2YXA8B']

"""'B00AYIDVLS': [2]--The Master Undone: An Inside Out Novella (Inside Out Series) Kindle Edition
by Lisa Renee Jones (Author), you can actually find the book by the asin"""
```

```
{'B0073VIZB0': [5],
 'B008BYG96Q': [5],
 'B008CD1H0O': [5],
 'B008GVC6SE': [5],
 'B009CE4TG6': [5],
 'B00AA46EDS': [4],
 'B00AEA7FWC': [5],
 'B00AQU7FTS': [5],
 'B00AYIDVLS': [2],
```

Potential Application & Further Development

1. Apply the same idea to specific books and do rating analysis of specific books
2. Link asin data to the current databases so that you can actually look into the books that got a bad rating from the user
   a. Maybe do some user's favorite/preferred genre analysis for recommender systems
3. Analyze if a reviewer is more objective when rating and update the rank/order of reviews accordingly
4. Recommend books based on similar reviews and ratings
   a. I.e. "User 1 gave same rating and review of book A, so you may like this book that User 1 liked"