Amazon reviews' sentiment

How do computers understand human languages? This project is an attempt at exploring natural language processing (NLP) through the public Amazon reviews dataset. Understanding sentiment embedded within human's writing can be very useful in many applications such as:

- Identify fake reviews
- Identify mistakes when a review is positive but the rating is low or vice versa.
- Extract features that impact to review's sentiment (what buyers care about most)

Data

Data for Amazon review is available publicly on Jianmo Ni's github webpage (https://nijianmo.github.io/amazon/index.html). This dataset includes reviews (ratings, text, helpfulness votes), product metadata (descriptions,
category information, price, brand, and image features), and links (also viewed/also bought graphs). There are several versions of the data available. The full raw version is very large (34gb) with a total of 233.1 million reviews. For the purpose of this project, a subset of the data is used: (5-core) in which all users and items have at least 5 reviews (total 75.26 million reviews).

Data Cleaning and Wrangling

Even if the smaller dataset (34 gb), the total amount of data needed to be processed is still more than a typical work computer using only panda can handle. For this project, pySpark is utilized to process a total of 30 json.gz files

Since data is separated into different categories, we need to combine them all into one set

```
#load all data
df = spark.createDataFrame([], schema)
df = df.withColumn('categories',lit(0))
for name in file_name:
  path = 'drive/MyDrive/Colab_Notebooks/Amazon_reviews/data/' + name
  #print('start: ', name[:-10])
  df_temp = spark.read.json(path, schema)
  df_temp = df_temp.withColumn('categories',lit(name[:-10]))
  df = df.union(df_temp)
  #print('done: ', name[:-10])
df = df.withColumn("vote",df.vote.cast('float'))
```

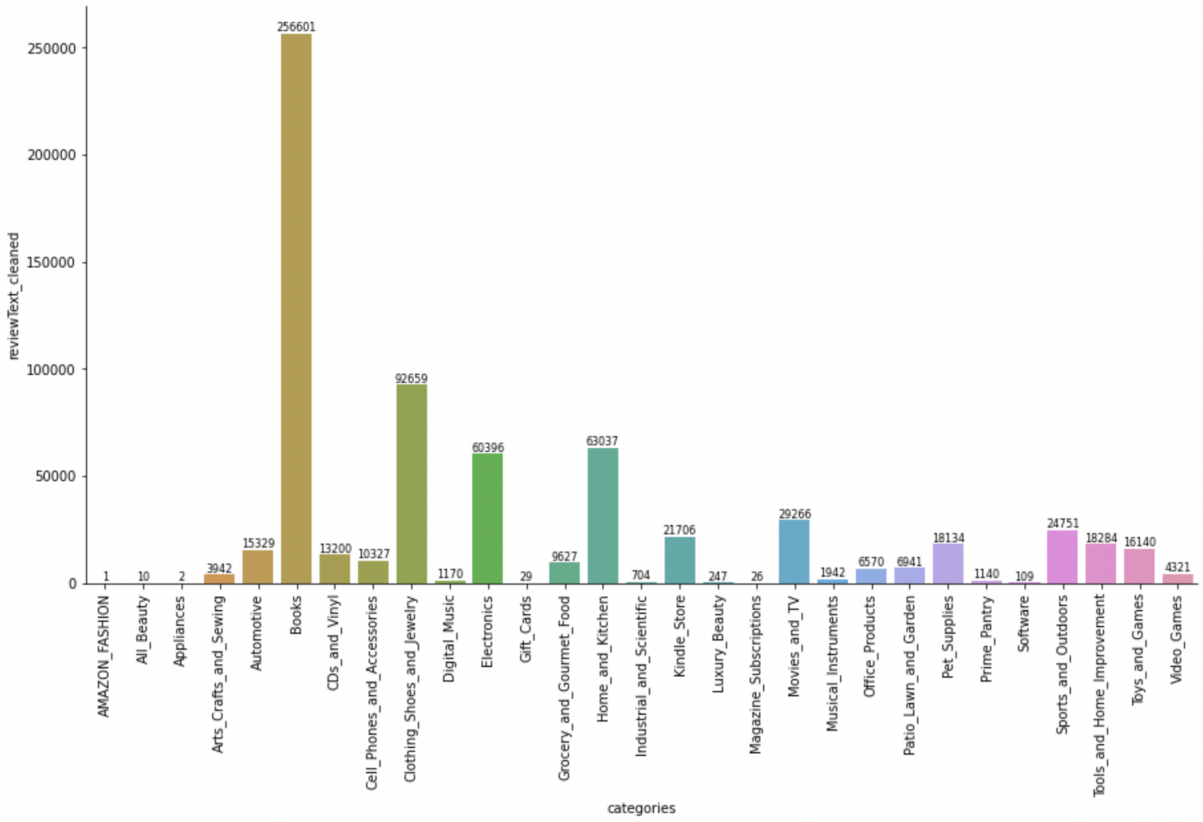
Below are the schema and a sample of our data

```
root
 |-- categories: string (nullable = true)
 |-- overall: double (nullable = true)
 |-- reviewText: string (nullable = true)
 |-- reviewerName: string (nullable = true)
 |-- summary: string (nullable = true)
 |-- verified: boolean (nullable = true)
 |-- vote: double (nullable = true)
```

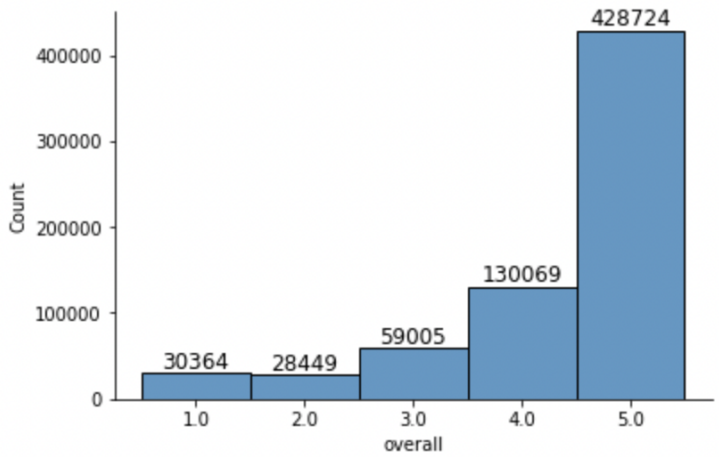
overall	reviewText	reviewerName	summary	verified	vote	categories
5.0	As someone who ha...	Loves Books in MD	Learn Adobe Photo...	false	null	Software
4.0	I've been running...	Mindcrime	Great product, bu...	false	14.0	Software
1.0	December 13, 2008...	James Smith	Amazon, PBJWORLD ...	false	5.0	Software
2.0	I have been a Qui...	Lance_big_daddy	Intuit has lost i...	false	31.0	Software
1.0	This is by far th...	Deimos	Garbage.....	false	null	Software

Exploratory Data Analysis

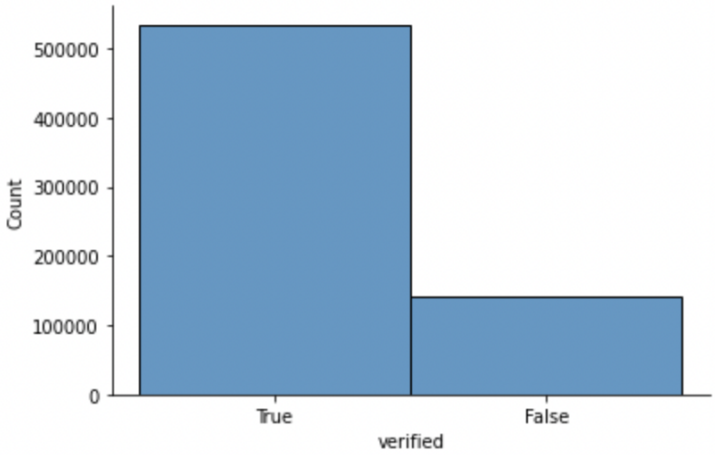
Let's take a look at the distribution of reviews in our data set by categories. As shown below, the books category has the most reviews, followed by clothing, shoes, and jewelry.



What's about the distribution of the overall score? Unsurprisingly, the rating is skewed to the right. THis could potentially caused imbalance issues in the modeling step.



Below is a plot of the number of reviews that are verified in comparison with reviews that are not. We can see that most reviews are actually from verified purchases.



Modeling

After some basic processing, our data set is reduced into 2 columns: text and label. The label is created based on the overall score. Ratings 1 and 2 are considered negative. Rating 4 and 5 are considered positive. Rating 3 is considered neutral and removed.

	text	label
0	must have for wow players best expansion for w...	positive
1	good game bought this for my son for his birth...	positive
2	stunningly beatiful but this game lacks one ma...	negative
3	five stars daughter addicted to game	positive
4	awesome game couldnt stop playing it fight nig...	positive

Reviews text is processed to remove any special character, single characters or multiple spaces.

```
processed_reviews = []

for review in range(0, len(X)):
    # Remove all the special characters
    processed_review = re.sub(r'\W', ' ', str(X[review]))

    # remove all single characters
    processed_review = re.sub(r'\s+[a-zA-Z]\s+', ' ', processed_review)

    # Substituting multiple spaces with single space
    processed_review = re.sub(r'\s+', ' ', processed_review, flags=re.I)

    processed_reviews.append(processed_review)
```

Afterward, feature vectors are created for each review using the TF-IDF approach. The classification model used for this project is random forest.

```
#create feature vectors containing TF-IDF values
tfidfconverter = TfidfVectorizer(max_features=500, min_df=5, max_df=0.7, stop_words=stopwords.words('english'))
X = tfidfconverter.fit_transform(processed_reviews).toarray()
```

```
#split train test dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
```

```
#train classification model
text_classifier = RandomForestClassifier(n_estimators=50, random_state=42)
text_classifier.fit(X_train, y_train)
```

The result from just a basic random forest is acceptable for positive sentiment. However, it doesn't perform as well for negative sentiment.

	precision	recall	f1-score	support
negative	0.76	0.31	0.44	11609
positive	0.93	0.99	0.96	111913
accuracy			0.93	123522
macro avg	0.85	0.65	0.70	123522
weighted avg	0.92	0.93	0.91	123522

Limitation and Future Work

The limitation of this project is the lack of access to more computing power. Without it, any work required is very time-consuming or out-of-reach which affects the ability to do proper hyperparameter tuning to improve the model's efficacy.
The next step of this project is to take the current out-of-the-box random forest model and improve his score. This could be achieved by:

- Undersampling the positive class or oversampling the negative to achieve a more balance dataset.
- Obtained computer power needed for hyperparameter tuning

Another approach for this project would be using state-of-the-art pre-trained NLP models such as BERT with TensorFlow framework.

Citation

Justifying recommendations using distantly-labeled reviews and fined-grained aspects
Jianmo Ni, Jiacheng Li, Julian McAuley
Empirical Methods in Natural Language Processing (EMNLP), 2019
pdf