Product Search Improvement in Amazon via Natural Language Processing

Team Members

Shreeti Upreti (upretiupreti@my.unt.edu)

Bijesh Patel Vachanni (bijeshpatelvachanni@my.unt.edu)

Github Repository

https://github.com/BIJESHPATEL369/CSCE-5290-Natural-Language-Processing-Project

Goals and Objectives:

Motivation

The motivation to work on this project has come from interests to work on solving real life problems. The Amazon KDD cup Challenge brings forth the opportunity to work on tasks that are implemented in the e-commerce website to enhance its predictive performance for product searches in an intuitive manner. The challenge does not only provide the dataset generated from the Amazon website but also guides the participants through the logical approach one can take to tackle problems of this nature. Moreover, the project also aligns with the concepts like text classification, minimum edit distance calculation, finding word similarity e.t.c that are being covered in the CSCE 5290 Section 001 - Natural Language Processing (Spring 2022 1) class.

Significance

The Amazon KDD cup Challenge as a project is a platform to apply Natural Language Processing concepts to practical usage. The challenge has three sub tasks, each dealing with different problems. Of the diverse tasks listed out for the challenge, two of the three tasks have been implemented with the possibility of applying the theoretical aspects learned thus far.

Task 1 : QUERY-PRODUCT RANKING
Task 2 : MULTICLASS PRODUCT CLASSIFICATION

Objectives

The objective is to find a better approach to information retrieval on semantic matching with user queries and relevant products. The products will be ranked based on the most to the least relevant products in the first task, given each Query as input. The second classifies the resultant products from the queries into four categories: exact, substitute, complement, and irrelevant. This ranking approach can leverage E-commerce to find the most profitable product ranking system to facilitate customers' search queries.

Features

Task 1:

The input will contain the user query, and the output will be products for each Query that will be ranked based on the most relevant ones.

Task 2:

The input will be product and Query pair along with product features like product title, description, brand, color, locale, etc., and the target labels will be following four categories:

- Exact: the most relevant product from the Query.
- Substitute: somewhat suitable product from the Query.
- Complement: the product will be mapped with other products to match the Query.
- Irrelevant: Product that is not relevant to the Query.

INCREMENT 1 : TASK 1 - Query-Product Search Ranking

Related Work

Retrieving products based on the queries entered and ranking them as per the relevance they show to the searched product is a practice that most e-commerce websites have been implementing and working on to make it more impactful. For instance, when a user types in “iphone” to the search bar, there are multiple products that match the keyword, like iphone charger, iphone case, iphone 11 pro max and the list might go on. It is not sufficient that the result shows the entire list of products that match the query but it is important that the most relevant products are shown first, i.e the iphone 11 pro max is shown first before the iphone accessories suggestions. Various ranking algorithms are available to achieve this kind of ranking for relevant products. For this task traditional Information Retrieval Models are being experimented with to achieve most relevant ranking.

The implementation of Information Retrieval on TREC test data set is documented in a medium article by Mayur Bhangale gives an insight to the models and concepts that can be leveraged to achieve ranking in documents. Based on what has been learned from the article, beside general natural language processing concepts used for cleaning and preprocessing the text data, for increment 1, Vector Space Model with Word2Vec is being used to find the ranks for results of query search results. The query and products are considered as a n-dimensional vector space with n being the number of index terms (product words or stems) , i.e matrix of term weights where rows are represented by texts that define the products and columns described by weights that have been assigned to the product text.

Dataset

The dataset that has been provided in the challenge is the real time data that has been collected from Amazon website. For this task, 2 different dataset has been provided. The training dataset that has been provided has the query text, the product id related to that query and label defining if the product that maps to the given product id is exact item, relevant item, substitute item or an irrelevant item. The data contained data collected from the US and Japan. The data collected from Japan is in the Japanese language while the one from the US is in English. Currently only US data is being worked upon.

	query_id	query	query_locale	product_id	esci_label
0	0	# 2 pencils not sharpened	us	B0000AQO0O	exact
1	0	# 2 pencils not sharpened	us	B0002LCZV4	exact
2	0	# 2 pencils not sharpened	us	B00125Q75Y	exact
3	0	# 2 pencils not sharpened	us	B001AZ1D3C	exact
4	0	# 2 pencils not sharpened	us	B001B097KC	exact

Further, another table with description of product with detailed information regarding the product such as product title, description, product brand and bullet points about them has been provided

	product_id	product_title	product_description	product_bullet_point	product_brand	product_color_name	product_locale
0	B0188A3QRM	Amazon Basics Woodcased #2 Pencils, Unsharpene...	NaN	144 woodcase #2 HB pencils made from high-qual...	Amazon Basics	Yellow	us
1	B075VXJ9VG	BAZIC Pencil #2 HB Pencils, Latex Free Eraser,...	BACK TO BAZIC Our go...	⭐ UN-SHARPENED #2 PREMIUM PENCILS. Each...	BAZIC Products	12-count	us
2	B07G7F6JZ6	Emraw Pre Sharpened Round Primary Size No 2 Ju...	<b>Emraw Pre-Sharpened #2 HB Wood Pencils -...	✓ PACK OF 8 NUMBER 2 PRESHARPENED BEGINNERS PE...	Emraw	Yellow	us
3	B07JZJLHCF	Emraw Pre Sharpened Triangular Primary Size No...	<b>Emraw Pre-Sharpened #2 HB Wood Pencils -...	✓ PACK OF 6 NUMBER 2 PRESHARPENED BEGINNERS PE...	Emraw	Yellow	us
4	B07MGKC3DD	BIC Evolution Cased Pencil, #2 Lead, Gray Barr...	NaN	Premium #2 HB lead pencils with break-resistan...	Design House	Gray	us

The two dataset provided, doesn't give enough information on the search relevance of a product that has been returned for the given query. So the two tables were merged to create a dataset that would provide sufficient information required to understand relevance of searched products and rank them accordingly.

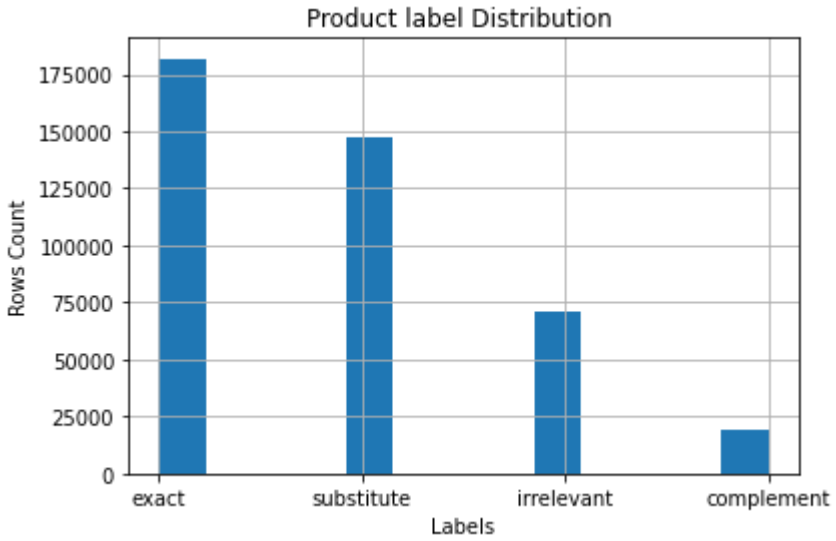
	query_id	query	product_id	product_title	product_brand	product_bullet_point	product_color_name	esci_label
0	0	# 2 pencils not sharpened	B0000AQO0O	Ticonderoga Beginner Pencils, Wood-Cased #2 HB...	Ticonderoga	Round wood pencil with latex-free eraser\nFini...	Yellow	exact
1	8799	pencils for kindergarteners	B0000AQO0O	Ticonderoga Beginner Pencils, Wood-Cased #2 HB...	Ticonderoga	Round wood pencil with latex-free eraser\nFini...	Yellow	exact
2	14844	#2 dixon oriole pencils not sharpened	B0000AQO0O	Ticonderoga Beginner Pencils, Wood-Cased #2 HB...	Ticonderoga	Round wood pencil with latex-free eraser\nFini...	Yellow	substitute
3	15768	#2 pencils with erasers sharpened not soft	B0000AQO0O	Ticonderoga Beginner Pencils, Wood-Cased #2 HB...	Ticonderoga	Round wood pencil with latex-free eraser\nFini...	Yellow	substitute
4	16972	classroom friendly supplies pencil sharpener	B0000AQO0O	Ticonderoga Beginner Pencils, Wood-Cased #2 HB...	Ticonderoga	Round wood pencil with latex-free eraser\nFini...	Yellow	irrelevant
5	0	# 2 pencils not sharpened	B0002LCZV4	TICONDEROGA Tri-Write Triangular Pencils, Stan...	Ticonderoga	Triangular shape to promote proper grip\nExclu...	Yellow	exact

Detail design of Features

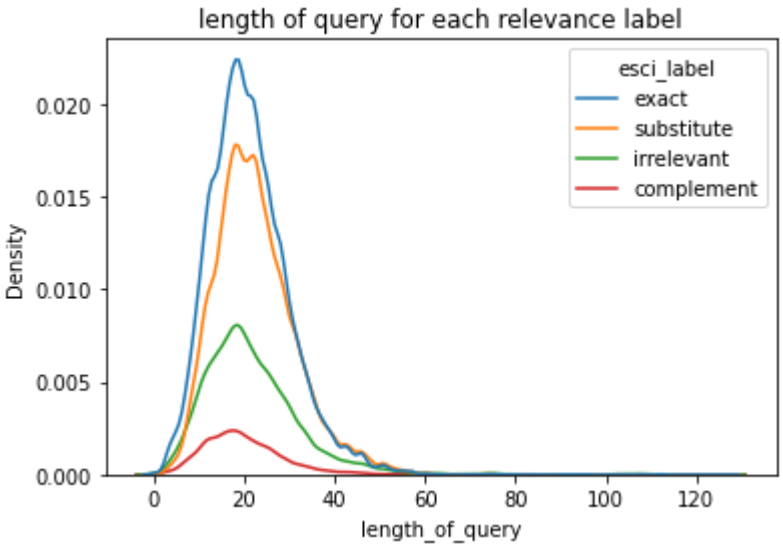
The dataset has a list of query typed by the real users to search for products in Amazon. The data has labels Exact, Substitute, Complement and Irrelevant. These labels are of much less significance in finding ranking for the query and products. The main features required to make the decision in raking is query and product information. For product information, we have product title, product brand name, product description and bullet points. All these features contribute to finding similarity in what has been typed as a query by the users. The listed features for product details were merged together to create a new feature product text using which the similarity is being calculated.

Vector Space Modeling is being used to represent features vectors to eventually calculate the similarity between those as scalar quantities.

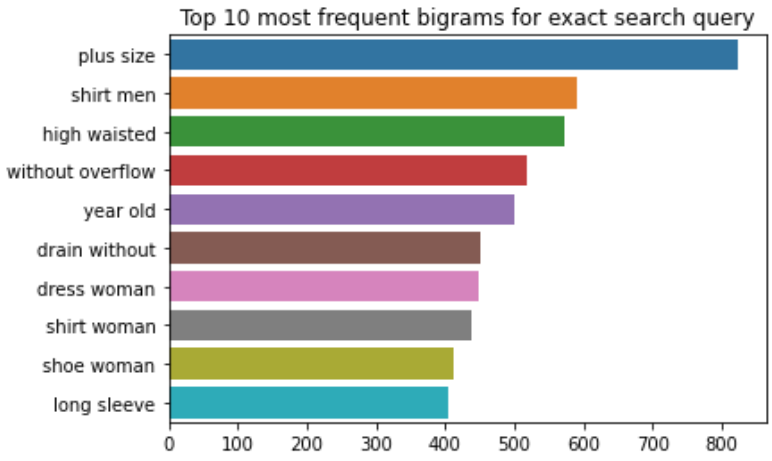
Analysis

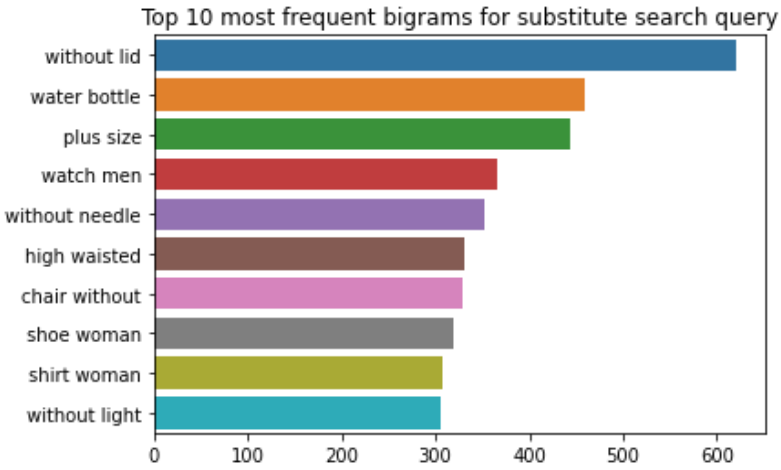


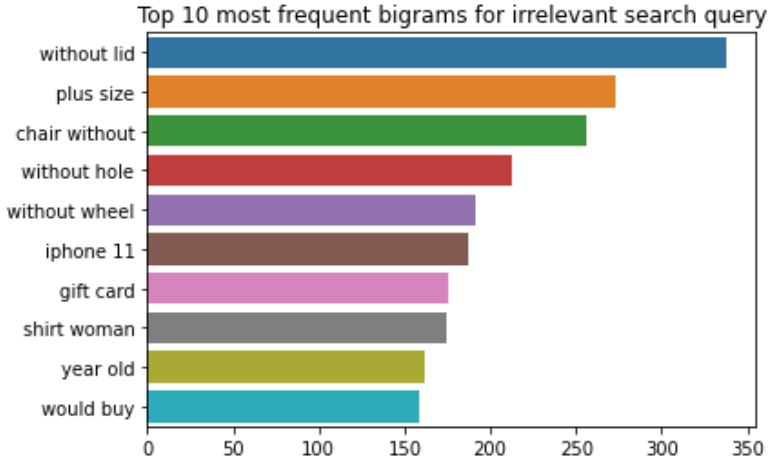
The class wise data distribution is not entirely balanced. The data distribution of data with class “complement” is way less in comparison to the other classes. However, beside the label, the similarity score is being used in case of ranking the product according to its relevance. So no oversampling or undersampling was done.

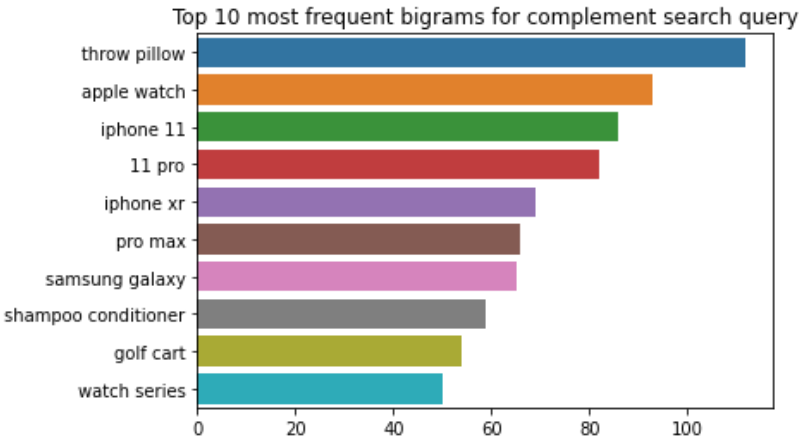


Here we check if query length has an influence on finding relevant products. Most queries have length between 10-30. The distribution is almost similar for all 4 relevance labels. The query length doesn't seem to have much meaningful impact





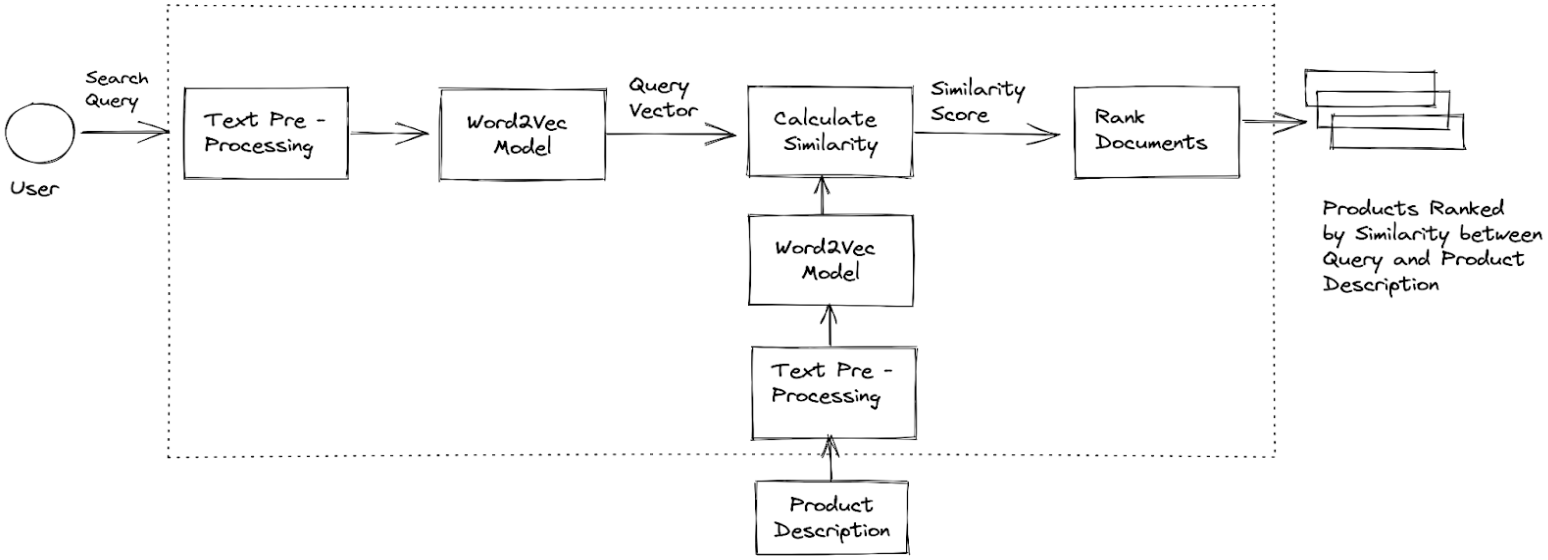




Top 10 bigrams from each class were being calculated to derive insights regarding the data. The bigrams don't look much different for the four classes to understand the relationship the query might have to do with the relevance.

As initially stated, The main features to be dealt with for making decisions about the ranking is query and product information.The product information features contribute to finding similarity in what has been typed as a query. Hence, product features were merged together to create a new feature to calculate similarity.

Implementation



The implementations have been done in following steps :

1. Data Filter : The data contained data collected from the US and Japan. The data collected from Japan is in the Japanese language while the one from the US is in English. Currently only US data is being worked upon. So the Japanese data was filtered out

2. Null Value Imputation: The data continued null values in product title and bullet points.The other data for “product bullet points” had elements of product title and product titles came close to what had been used as queries. So, the null values for product title were filled with corresponding queries and product bullet points with corresponding product titles. Also, for brand name, according to the data trend the first work in the product generally was the band's name that manufactured the product. So the null spaces for brand name were filled with the first word of the product title

3. Data Cleaning : The goal here is to find the similarity between the query used to search the product and the actual text used for the product. In order to do this it is necessary to clean text so that a robust relationship can be established between query and product title.The measuring units in the products were not uniform. For example, for unit pounds some titles used ‘lb’ while the others use ‘lbs’ or ‘pounds’.

This non uniformity made it hard to establish relationships between sentences to compare similarities. Similarly spacing was added between numeric values and alphabets. In instances where “5vitamins” and “5 vitamins” refer to same product, no similarity can be found in them due to the spaces. So the numbers and alphabets were spaced in such cases. Another case that was taken care of was spaces before and after “-”s and “\”s to resolve similar cases. Beside these case specific cleaning, texts were all changed to lowercase and extra white spaces were removed to get clean data before we can perform further operations

4. Lemmatization and stop words removal : The stop words were removed and the query and product text were lemmatized in order to clean the text

5. Spelling correction : When users type a query, it is likely for them to make typographical errors. The spellings were corrected using “SpellChecker” library from python

6. Vectorization : Since the data is text data, it needs to be represented as numbers i.e in mean vector form. Word2vec is being used here to achieve that as it converts words into vectors while keeping the semantic and syntactic relationship of words intact.

Preliminary Results

	query_id	query	product_id	esci_label	product_text
346888	14575	[-0.14417554, -0.0032218695, -0.03756547, 0.10...	B06WGSJG33	2	[0.2981141596452496, 0.30604560257426217, 0.21...
162442	5449	[0.4713722, 0.11496067, -0.45415765, -0.220027...	B07413JWLD	0	[0.38019696830769234, 0.2695880662354785, -0.0...
349662	14702	[-0.06720107, -0.18399131, -0.08664151, -0.330...	B07CX6Z6JT	0	[0.22638627440153833, 0.1771063471363805, 0.02...
240969	14114	[0.13409637, 0.74019253, -0.61410445, -0.36707...	B00631YUEM	2	[0.36237794977995, 0.3217945276051556, 0.03330...
264607	10187	[0.25220913, 0.76465404, 0.413657, -0.21542335...	B07PXGVRMD	3	[0.2669751903862871, 0.44798220351913304, 0.06...

	query_id	query	product_id	esci_label	product_text	similarity
5022	63	[-0.050902568, -0.18287396, 0.11213769, 0.1374...	B00AZWYUA4	1	[0.2025589897528482, 0.23592547450674656, 0.10...	0.632379
5020	63	[-0.050902568, -0.18287396, 0.11213769, 0.1374...	B004AMEOZG	2	[0.08380478378161116, 0.07156367621437207, -0....	0.632379
5017	63	[-0.050902568, -0.18287396, 0.11213769, 0.1374...	B0031450XY	3	[0.1407073632172678, 0.18506185337966363, 0.10...	0.632379
5007	63	[-0.050902568, -0.18287396, 0.11213769, 0.1374...	B0006HXQXA	1	[0.0776666815731914, 0.22133194655614388, 0.15...	0.632379
5013	63	[-0.050902568, -0.18287396, 0.11213769, 0.1374...	B001T6QCJC	3	[0.1758094176033482, 0.22056496028155886, 0.09...	0.632379
4974	63	[-0.050902568, -0.18287396, 0.11213769, 0.1374...	B0016P2A5G	0	[0.13634659580322853, 0.16153233425666844, 0.1...	0.632379
5015	63	[-0.050902568, -0.18287396, 0.11213769, 0.1374...	B002VKWSYI	3	[0.06784091120282323, 0.23914952195412936, 0.1...	0.632379

```
Mean Average Precision=> 0.5845550106384466
```

Word2Vec model was trained using a corpus that was created from the dataset to convert the text features into their vector forms, similarity scores for each pair of query and product text were calculated using “cosine_similarity” which is further being used to rank the documents. The metrics used to evaluate the result of the ranking is Mean Average Precision (MAP). When no relevant document can be retrieved the MAP value is 0. For the value information that is needed, the MAP calculates the average area under the precision-recall curve for the given queries. Thus far the Mean Average Precision that has been achieved is approximately 58%. This value can be further increased by tuning the model for higher accuracy in ranking the documents which will further be explored in Increment 2.

INCREMENT 1 :Task 2 : MULTICLASS PRODUCT CLASSIFICATION

Related Work:

The approach for transforming the text into a vector representation uses the Query2Vec word embedding algorithm. The preprocessed queries and related features are transformed to vector embedding by vectorizing the phrase with the similarity between query and feature using the cosine similarity method. This algorithm is that it can convert the vector representation with low dimensionality by compressing the massive amount of text into small dimensions. Another approach for word embedding is FAISS, which finds the similarity by initializing the clusters and creating a matrix of vectors on them.

The fundamental algorithms for text classification problems are Linear classifiers, naive Bayes classifiers, Support Vector Machine, neural networks, etc. The model can only take numerical data for classification problems, so the encoded labels are converted from string to numerical representation. There are numerous ways of encoding the string label to numerical values, one of the approaches is by label encoder, which converts these labels to binary format. These transformed text and labels are trained on machine learning algorithms to predict the labeled classes.

Dataset:

The Dataset attempts to solve the semantic matching problem for shopping queries and relevant product search on manually annotated classes: Exact (E), Substitute (S), Complement (C), and Irrelevant (I). The Data are available in three multilingual languages: English, Japanese and Spanish. The data set comes in two sets: a training set and a product catalog set—the training set has Query, Product IDs, and target labels. The product catalog set has Product-related information with Product, Product Title, product brand, product description, product bullet point, product color, and product locale.

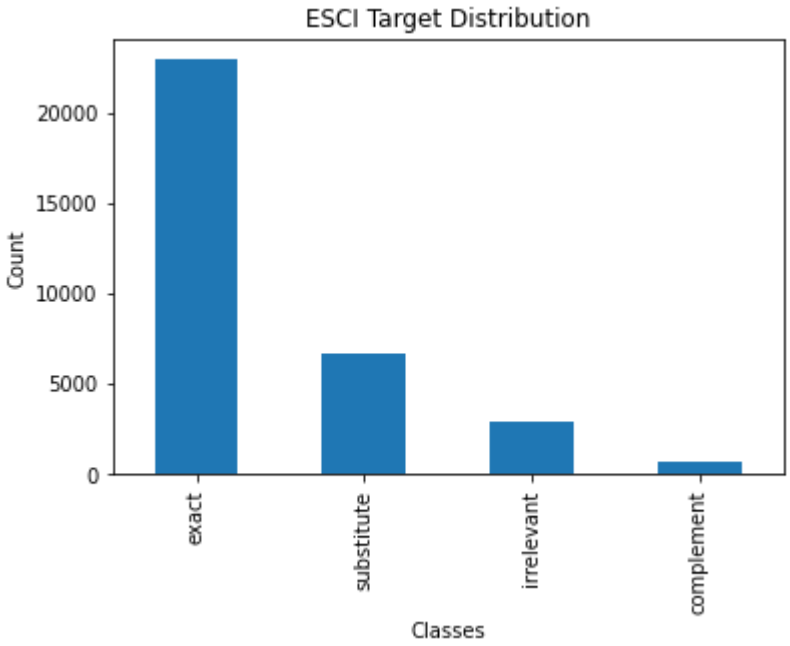
Detail Design of features:

The main features for this task are query and product metadata, and the product metadata contains seven fields: product id, title, description, bullet point, brand, color name, and locale. There are two different locales US and Japan, but we will consider the US locale only.

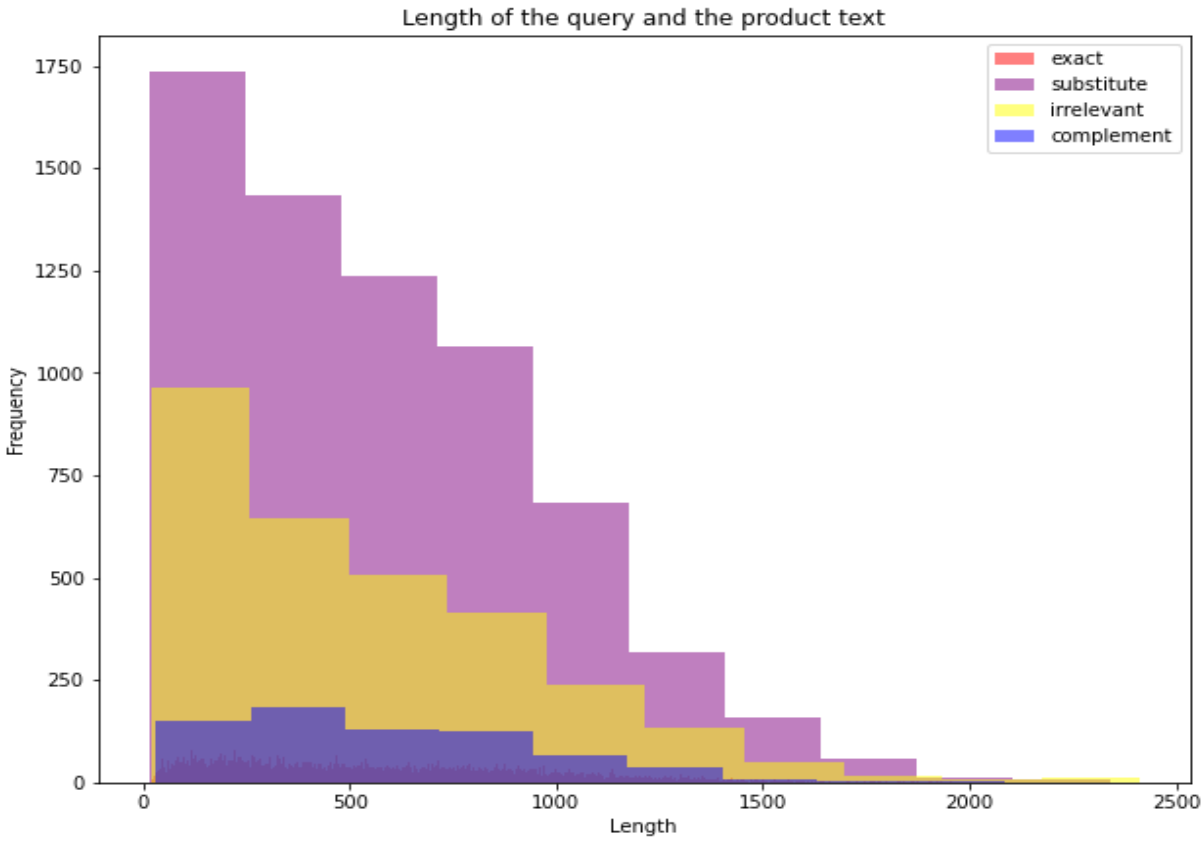
We have concatenated the product metadata to locate the similarity between the query and the product metadata into one text column. The product metadata has too many missing values resolved by doing this. Later join this single column on a product data frame with a train data frame based on product id. In feature pre-processing, we performed the following task: removing punctuation, converting uppercase letters to lower case letters, using poster stemmer for stemming words from NLTK and removing all the English stop words using NLTK corpus. The model can only take one input feature for the traditional text classification approach. Therefore, the query and product text data strings are concatenated into one column ‘product_text’.

To find the variation between text length against the target labels, we create a new column, ‘length_query_product_text', containing the size of the input feature.

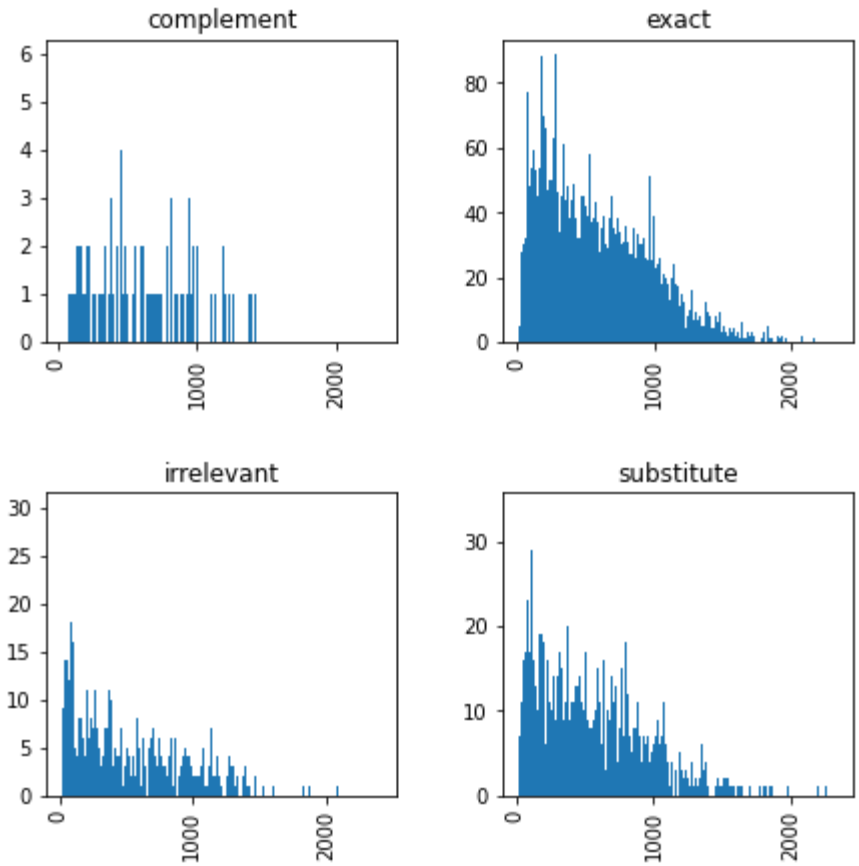
Analysis:



The target distribution has unbalanced classes, and the Complement class is significantly less in number, whereas the Exact Class is very high. As this is a classification problem to deal with this problem, the micro F1 score is applied to predict the model's accuracy during evaluation.



The Substitute and Irrelevant classes have the highest character distribution, whereas the exact is very low. To resolve this problem, we can remove the product description data with very long and unrelated information with product and query search text.



The histogram of each target length distribution illustrates that the Irrelevant class is a Right Skewed distribution with a high peak on the right side of the plot, so the extensive amount of data tends to fall on one side. In contrast, the other three classes do not have this problem.

Implementation:

Text Representation:

Machine learning algorithms can only take input features in numerical format, so the string text will be converted to numerical format using text embedding methods. For task 2, we have used the following Text embedding technique:

TF-IDF:

TF-IDF is one of the text vectorizer techniques that is relatively identical to the bag of words technique where it finds the relevance of the word from a given text. In TF-IDF, the term frequency is based on the count of words in text to the number of words in the text, and the inverse document frequency is to find the more importance of a word in the given

text. Column Transformer is used to create the vectorizer of Query and Product pair, by specifying the no of features the training process can be made faster. We have encoded the labels using the Label encoder from Scikit learn before passing to the model.

Universal Sentence Encoder:

The USE sentence embedding technique converts text features into vectors utilized in text classification. The Tensor-flow hub provides the pre-trained embedding methods. The transformer encoder has higher accuracy than the DAN encoder, but the transformer is very slow in performance, whereas DAN is vice versa. We load Universal sentence encoder multilingual three from the tensor hub. To make the training process easy, we split the training and testing data into 500 batches of array split. After the vector conversation, we have trained these features and target labels of the training data set to a random forest classifier model using the fit function. Later, the testing data set attributes are passed to the above-trained model, which predicts the target labels. Finally, get accuracy based on the ground truth and predicted labels.

Model:

For Task 2, based on the high dimensional feature vectors to simplify the training process, we are using a random forest classifier to test the performance of the TF-IDF, Universal Sentence Encoder embedding techniques. The random forest classifier can perform well with massive data, which is also prevalent for classification tasks. The model is trained on TF-IDF and Universal Sentence encoder feature vectors, and the model predictions on encoded target labels.

Evaluation:

The classification task is evaluated based on the f1 score, in task 2 the target classes are unbalanced so we are using micro averaging f1 score to evaluate the model accuracy.

F1 = 2 precision * recall/ precision + recall

Preliminary Results:

The accuracy for TF-IDF and Random Forest:

Model Evaluation

```
[64]  1 from sklearn.metrics import accuracy_score as acc
      2
      3 accuracy_td_idf = acc(y_test, y_pred)
      4 print("Micro F1 Score Accuracy for TF-IDF trained on random forest model: ", accuracy_td_idf)

Micro F1 Score Accuracy for TF-IDF trained on random forest model:  0.6809659517024149
```

The accuracy for Universal Sentence Encoder and Random Forest model:

Model Evaluation

```
[77]  1 from sklearn.metrics import accuracy_score as acc
      2
      3 acc_use = acc(y_test, y_pred_embedded)
      4 print("Micro F1 Score Accuracy for Universal Sentence encoding trained on random forest: ", accuracy_td_idf)

Micro F1 Score Accuracy for Universal Sentence encoding trained on random forest:  0.6809659517024149
```

The accuracy tends to be the same for both the TF-IDF and Universal sentence encoder when trained on a random forest model of 68 %.

Project Management

Implementation status report

Work completed:

Description:

Task 1

The data has been pre-processed and manipulated to best suit the models used to rank the products. The data has been trained using the Vector space model with Word2Vector. The similarity scores for query and product text is being used to determine the ranking of the products.

Task 2

We have implemented the feature selection, pre-processing of the features, data cleaning, data visualization, various techniques to represent the text, model training, and accuracy prediction.

Responsibility :

- Task 1 : Query-Product Ranking (Shreeti Upreti)
- Task 2 : Multiclass Product Classification (Bijesh Patel Vachanni)

Contributions

- Shreeti Upreti : 50%
- Bijesh Patel Vachanni : 50%

Work to be completed:

Description

Task 1

The evaluation metric that is being used thus far is the Mean Average Precision which is 61%. Now further work will be focused on experimenting with other machine learning algorithms to train the data and to tuning the algorithms to achieve better performance.

Task 2

Training and tuning the features with different models to find the best accuracy. Comparing the accuracy with the state-of-art BERT model.

Responsibility

- Task 1 : Query-Product Search Ranking performance tuning and experimenting with Longform model (Shreeti Upreti)
- Task 2 : Multiclass Product Classification (Bijesh Patel Vachanni)

Issues/Concerns

1. Gaining optimum accuracy for the model with lesser runtime.
2. Handling overfitting/underfitting issues

INCREMENT 2 : TASK 1 - Query-Product Search Ranking

Introduction

The task is related to ranking the product options that are retrieved once a query is placed in a search engine. Once the user inputs a query, the search engine looks for products that are closely related to the texts in input. Based on the extent of similarity between query and product title of the results, they are ranked and displayed in order of their ranking.

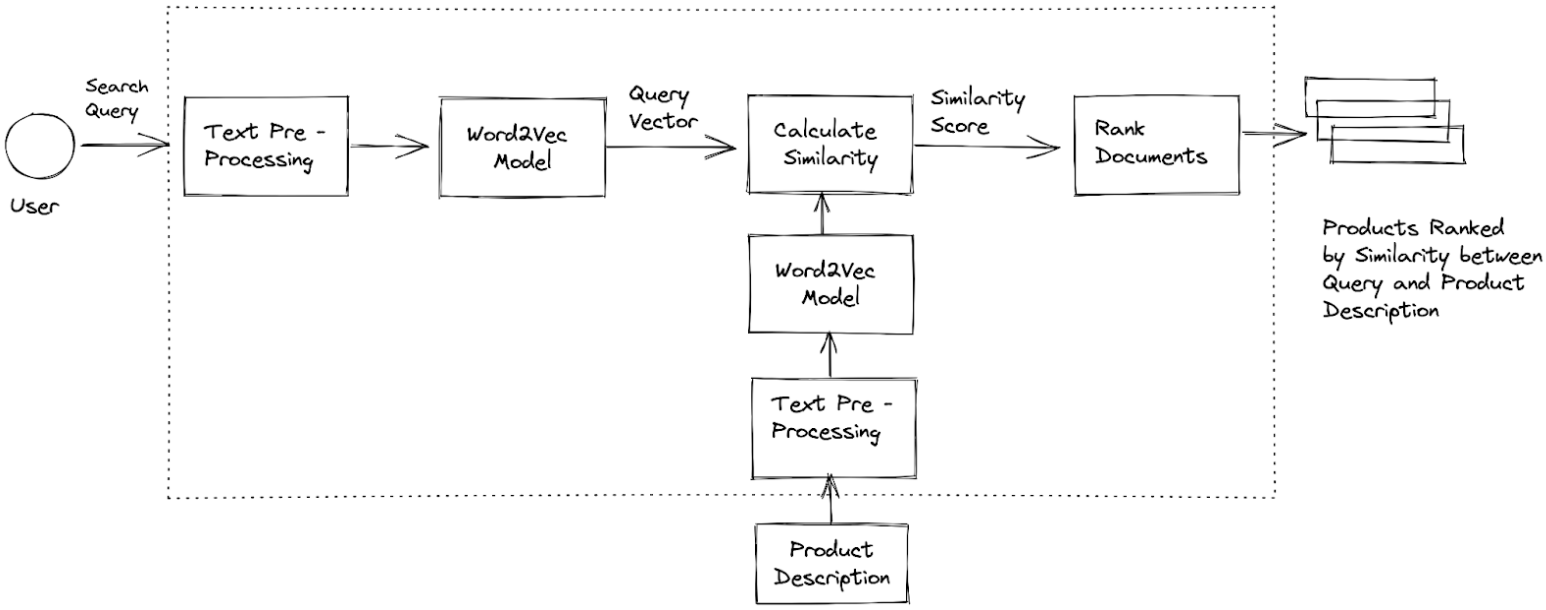
The model implementation, tuning and retrieval of information is being done in increment 2.

Background

An Article from medium was referred to for references in retrieving information and ranking them in terms of relevance. The article implements information retrieval from TREC and ranks them. It also discusses the concept of Learning to Rank. It also further discusses the model that can be used to achieve the results and metrics that can be used to evaluate implementations .

From what has been understood through the article, in increment 2 the Vector Space Model was explored in detail to achieve ranked results. The embeddings are calculated using Word2Vec for queries and products. Further, a similarity score for the two was calculated to rank the products.

Model



Vector space models are being used to retrieve products based on their ranks. Vector space modeling is an unsupervised method which is being used here in combination with the concept of similarity. It assumes that the relevant relationship between a query-text and product text has direct relation to their vector similarity score. This means that the query-product pairs with higher similarity score are more relevant than the ones with lower score.

Dataset

The dataset that has been provided by the Amazon KDD cup challenge for this task has two sets of data. The first data has information that is helpful in mapping query to its corresponding products. It provides columns with information regarding query id, query text, product id and their label denoting how close the query is to the corresponding product and label defining whether the query-product pair is exact, relevant, substitute or an irrelevant item.

	query_id	query	query_locale	product_id	esci_label
0	0	# 2 pencils not sharpened	us	B0000AQO0O	exact
1	0	# 2 pencils not sharpened	us	B0002LCZV4	exact
2	0	# 2 pencils not sharpened	us	B00125Q75Y	exact
3	0	# 2 pencils not sharpened	us	B001AZ1D3C	exact
4	0	# 2 pencils not sharpened	us	B001B097KC	exact

Another dataset that has been given has product description with other related details like product's title, description, brand it is related to and bullet points regarding the same products.

	product_id	product_title	product_description	product_bullet_point	product_brand	product_color_name	product_locale
0	B0188A3QRM	Amazon Basics Woodcased #2 Pencils, Unsharpene...	NaN	144 woodcase #2 HB pencils made from high-qual...	Amazon Basics	Yellow	us
1	B075VXJ9VG	BAZIC Pencil #2 HB Pencils, Latex Free Eraser,...	BACK TO BAZIC Our go...	⭐ UN-SHARPENED #2 PREMIUM PENCILS. Each...	BAZIC Products	12-count	us
2	B07G7F6JZ6	Emraw Pre Sharpened Round Primary Size No 2 Ju...	<b>Emraw Pre-Sharpened #2 HB Wood Pencils -...	✓ PACK OF 8 NUMBER 2 PRESHARPENED BEGINNERS PE...	Emraw	Yellow	us
3	B07JZJLHCF	Emraw Pre Sharpened Triangular Primary Size No...	<b>Emraw Pre-Sharpened #2 HB Wood Pencils -...	✓ PACK OF 6 NUMBER 2 PRESHARPENED BEGINNERS PE...	Emraw	Yellow	us
4	B07MGKC3DD	BIC Evolution Cased Pencil, #2 Lead, Gray Barr...	NaN	Premium #2 HB lead pencils with break-resistan...	Design House	Gray	us

Given that the two dataset do not provide enough information, individually, to make a decision on relevance, the two tables were merged into a single dataset that could provide sufficient information required to understand relevance of searched products and rank them accordingly.

	query_id	query	product_id	product_title	product_brand	product_bullet_point	product_color_name	esci_label
0	0	# 2 pencils not sharpened	B0000AQO0O	Ticonderoga Beginner Pencils, Wood-Cased #2 HB...	Ticonderoga	Round wood pencil with latex-free eraser\nFini...	Yellow	exact
1	8799	pencils for kindergarteners	B0000AQO0O	Ticonderoga Beginner Pencils, Wood-Cased #2 HB...	Ticonderoga	Round wood pencil with latex-free eraser\nFini...	Yellow	exact
2	14844	#2 dixon oriole pencils not sharpened	B0000AQO0O	Ticonderoga Beginner Pencils, Wood-Cased #2 HB...	Ticonderoga	Round wood pencil with latex-free eraser\nFini...	Yellow	substitute
3	15768	#2 pencils with erasers sharpened not soft	B0000AQO0O	Ticonderoga Beginner Pencils, Wood-Cased #2 HB...	Ticonderoga	Round wood pencil with latex-free eraser\nFini...	Yellow	substitute
4	16972	classroom friendly supplies pencil sharpener	B0000AQO0O	Ticonderoga Beginner Pencils, Wood-Cased #2 HB...	Ticonderoga	Round wood pencil with latex-free eraser\nFini...	Yellow	irrelevant
5	0	# 2 pencils not sharpened	B0002LCZV4	TICONDEROGA Tri-Write Triangular Pencils, Stan...	Ticonderoga	Triangular shape to promote proper grip\nExclu...	Yellow	exact

Implementation

Algorithms / Pseudocode

STEP 1 : Merge two datasets to create a single data set with detailed information necessary to create relationship between query and products

STEP 2 : Filter data to only keep information from USA

STEP 3 : Impute null values with relevant values

STEP 4 : Clean text data from extra white spaces. Chance texts to lower cases. Standardize measurement units.

STEP 5 : Remove stop words and lemmatize the texts in query and product texts using nltk library

STEP 6 : Correct spelling using python library

STEP 7 : Vectorize text data using Word2Vec

STEP 8 : Calculate similarity score for query-product pair

STEP 9 : Create function to return relevant product based on their ranks upon passing query as function parameter

STEP 10 : Evaluate results.

Explanation of implementation

The implementations have been done in following steps :

1. Vectorization : Since the data is text data, it needs to be represented as numbers i.e in mean vector form. Word2vec is being used here to achieve that as it converts words into vectors while keeping the semantic and syntactic relationship of words intact.

2. Similarity calculation : Using the word vectors that was initially calculated, cosine similarity for each of the query and product details pair was calculated. This similarity score is being used to decide the relevance of the product results based on the query.

3. Model : Unsupervised learning via Vector space model with Word2vec was being used to calculate vector embeddings. The corpus to train the Word2vec was created using the texts extracted from query and product columns from the dataset.

   After having vectorized the words, the similarity score for query-product pair texts was calculated to rank the products based on the similarity .

4. Evaluation : The metrics used to evaluate the result of the ranking is Mean Average Precision (MAP). When no relevant document can be retrieved the MAP value is 0. For the value information that is needed, the MAP calculates the average area under the precision-recall curve for the given queries.Mean Average Precision of 61% was achieved.

Results

	query_id	query	product_text	similarity
0	6201	japan deep frying pan	bushcraft takibi deep frying pan bushcraft tak...	0.734204
1	3103	copper chef 9 5 inch square diamond pan lid	copper chef 10 inch diamond fry pan round fryi...	0.713693
2	3103	copper chef 9 5 inch square diamond pan lid	copper chef 9 5 inch diamond fry pan square fr...	0.703848
3	10946	stainless steel pot without handle	avacraft 18 10 tri ply stainless steel frying ...	0.699248
4	3103	copper chef 9 5 inch square diamond pan lid	copper chef 12 inch diamond fry pan square fry...	0.696607
5	3103	copper chef 9 5 inch square diamond pan lid	14 inch 1 multi use copper chef wonder cooker ...	0.692020
6	15279	complicated katie lee	healthy sheet pan cookbook satisfying one pan ...	0.681805
7	15962	10 skillet without aluminum	avacraft 18 10 stainless steel frying pan lid ...	0.681404
8	11074	stonelainy	small ceramic rectangular dish baking dish han...	0.680373
9	206	1 2 quarter pan without lid lightweight	avacraft 18 10 stainless steel frying pan lid ...	0.678168

Similarity score for each query-product pair was calculated using the cosine similarity to check how closely related the query are to the product results. Based on the similarity score the product results are ranked in order of their relevance.

```
query = 'air conditioner'
print('The most relevant products to the search "{}" are:'.format(query))
for i, product in enumerate(main(query, 10)):
  print(i+1, product)
```

```
The most relevant products to the search "air conditioner" are:
1 Danby 10,000 BTU Window Air Conditioner with Remote
2 Danby 15,000 BTU Window Air Conditioner with Remote
3 LG Electronics 7,000 BTU Window Air Conditioner with Cool, Heat and Remote
4 LG Electronics 7,000 BTU Window Air Conditioner with Cool, Heat and Remote
5 Danby 12,000 BTU Window Air Conditioner with Remote
6 Whynter 14,000 BTU Portable Air Conditioner with Dehumidifer and Remote
7 LG Electronics 12,000 BTU Window Air Conditioner with Cool, Heat and Remote
8 Honeywell 14,000 BTU Portable Air Conditioner with Remote Control in Black and Silver
9 LG Electronics 23,500 BTU Window Air Conditioner with Cool, Heat and Remote
10 LG Electronics 7,500 BTU 115-Volt Window Air Conditioner with Cool, Heat and Remote


oking for queries with incorrect spelling
```

Having done that, whenever a query is passed in to look for a product, the model now returns the top 10 most relevant products in ascending order of their relevance. The same results were achieved when typing any query with typo too. The query passes through a spell checker that attempts to find the closest word to the query with typographical error and corrects them before looking for a product.

```
query = 'air conditioner'
print('The most relevant products to the search "{}" are:'.format(query))
for i, product in enumerate(main(query, 10)):
  print(i+1, product)
```

```
The most relevant products to the search "air conditioner" are:
1 Danby 10,000 BTU Window Air Conditioner with Remote
2 Danby 15,000 BTU Window Air Conditioner with Remote
3 LG Electronics 7,000 BTU Window Air Conditioner with Cool, Heat and Remote
4 LG Electronics 7,000 BTU Window Air Conditioner with Cool, Heat and Remote
5 Danby 12,000 BTU Window Air Conditioner with Remote
6 Whynter 14,000 BTU Portable Air Conditioner with Dehumidifer and Remote
7 LG Electronics 12,000 BTU Window Air Conditioner with Cool, Heat and Remote
8 Honeywell 14,000 BTU Portable Air Conditioner with Remote Control in Black and Silver
9 LG Electronics 23,500 BTU Window Air Conditioner with Cool, Heat and Remote
10 LG Electronics 7,500 BTU 115-Volt Window Air Conditioner with Cool, Heat and Remote
```

cking for queries with incorrect spelling

```
query = 'aer conditoner'
print('The most relevant products to the search "{}" are:'.format(query))
for i, product in enumerate(main(query, 10)):
  print(i+1, product)
```

```
The most relevant products to the search "aor condiotioner" are:
1 Danby 10,000 BTU Window Air Conditioner with Remote
2 Danby 15,000 BTU Window Air Conditioner with Remote
3 LG Electronics 7,000 BTU Window Air Conditioner with Cool, Heat and Remote
4 LG Electronics 7,000 BTU Window Air Conditioner with Cool, Heat and Remote
5 Danby 12,000 BTU Window Air Conditioner with Remote
6 Whynter 14,000 BTU Portable Air Conditioner with Dehumidifer and Remote
7 LG Electronics 12,000 BTU Window Air Conditioner with Cool, Heat and Remote
8 Honeywell 14,000 BTU Portable Air Conditioner with Remote Control in Black and Silver
9 LG Electronics 23,500 BTU Window Air Conditioner with Cool, Heat and Remote
```

Word2Vec model was trained using a corpus that was created from the dataset to convert the text features into their vector forms, similarity scores for each pair of query and product text were calculated using “cosine_similarity” which is further being used to rank the documents. The metrics used to evaluate the result of the ranking is Mean Average Precision (MAP). When no relevant document can be retrieved the MAP value is 0. For the value information that is needed, the MAP calculates the average area under the precision-recall curve for the given queries.

The Mean Average Precision that has been achieved is approximately 61% which is better that what we initially received, that is 58%.

```
Mean Average Precision=> 0.6119489025339347
```

INCREMENT 2 :Task 2 : MULTICLASS PRODUCT CLASSIFICATION

Methods:

Bert Model:

We choose the model as a Bidirectional encoder representation from transformers (Bert), considered the state-of-the-art model for all-natural language processing tasks. The model is implemented on attention mechanisms to find the similarity between the words for a given text with extensive parameters. The encoder path computes the vector representation to the given context, and the decoder encounters the similarity in the given sequence.

Albert Model:

Albert's model is the most robust model that compresses to very minimal parameters with very high performance compared to other models. This model enacts on the encoder and decoder paths. The encoder path has attention to adding the vector representation to the given context, and the decoder path has a self–attention mechanism to compute the sequence similarity. The model structure has multiple blocks with multi-head attention modules and deep feedforward networks—the factorization of embedding technique and cross-layer parameter sharing to reduce the parameters.

Implementation:

We trained the Bert model on the same hyper-parameters previously pre-trained on the model in the training process. We mainly used the same hyper-parameters once used on the pre-trained model by altering the most contributing hyper-parameters to fine-tune the Albert model. The major contributing hyperparameters are Learning rate of 0.00002, bias, gamma, beta, weight decay rate of 0.01, epoch of 4, and batch size of 64. There were 109, 458, 316 training parameters for the Bert model and 11,686,660 training parameters for the Albert model. Bert model's training time and GPU are approximately 2 times the Albert model.

Results:

Classification report:

```
1 label_bert, prediction_bert = testing(model_bert,test_loader_bert,optimizer_bert)
2
3 print(classification_report(label_bert, prediction_bert, target_names=['exact','substitute', 'irrelevant', 'complement']))
```

```
              precision    recall  f1-score   support

       exact       0.87      0.91      0.89     34334
  substitute       0.65      0.63      0.64     10147
  irrelevant       0.67      0.51      0.58      4403
  complement       0.52      0.38      0.44      1116

    accuracy                           0.80     50000
   macro avg       0.67      0.61      0.64     50000
weighted avg       0.80      0.80      0.80     50000
```

Figure : Bert Classification Report

```
1 label_albert, prediction_albert = testing(model_albert,test_loader_albert,optimizer_albert)
2
3 print(classification_report(label_albert, prediction_albert , target_names=['exact','substitute', 'irrelevant', 'complement']))
```

```
              precision    recall  f1-score   support

       exact       0.82      0.88      0.85     34334
  substitute       0.48      0.45      0.46     10147
  irrelevant       0.45      0.35      0.40      4403
  complement       0.00      0.00      0.00      1116

    accuracy                           0.73     50000
   macro avg       0.44      0.42      0.43     50000
weighted avg       0.70      0.73      0.71     50000
```

Figure: Albert Classification Report

The model comparison on the classification report illustrates that Bert model F1 score on the Exact Class has a high true-positive prediction. The other three classes have reasonably average true-positive predictions. In contrast, Albert model F1 score on Exact Class has very high predictions, Substitute, and Irrelevant Class has moderate predictions, and complement class has no prediction. The average micro F1- score for the model trained on Bert computed 80% accuracy, and the model trained on Albert computed 73% accuracy. The Bert model produced a better prediction than the Albert model from the above result.

Project Management

Work completed:

Description:

Task 1

A model has been implemented using unsupervised learning via the Vector Space Model. The embeddings are calculated using Word2Vec for queries and products. Further, a similarity score for each of the query-product pairs was calculated using cosine similarity. Based on this relevance score, the products were ranked.

Task 2

We have Fine-tuned pre-trained Bert and Albert models to get the best accuracy. According to the micro average F1 scores of the model, the prediction of the Bert model is better than the Albert model.

Responsibility :

- Task 1 : Finding Query-Product Ranking for given queries (Shreeti Upreti)
- Task 2 : Multiclass Product Classification (Bijesh Patel Vachanni)

Contributions

- Shreeti Upreti : 50%
- Bijesh Patel Vachanni : 50%

References/Bibliography

1. https://stackoverflow.com/questions/19841535/python-matplotlib-venn-diagram
2. https://www.quora.com/How-do-I-extract-intent-from-a-search-query-using-NLP
3. https://www.quora.com/How-does-NLP-enhance-search-quality
4. https://medium.com/swlh/relevance-ranking-and-search-a98b35ebc7b3
5. https://medium.com/coursera-engineering/query2vec-2f6070083bda
6. https://www.cs.uic.edu/~cornelia/papers/iscram_asian18.pdf
7. https://www.iieta.org/journals/ria/paper/10.18280/ria.350404
8. https://huggingface.co/cross-encoder/ms-marco-MiniLM-L-12-v2
9. https://arxiv.org/pdf/2004.03705.pdf
10. https://pypi.org/project/sentence-transformers/
11. https://huggingface.co/docs/transformers/model_doc/bert
12. https://huggingface.co/docs/transformers/model_doc/albert