# E-commerce Product Classification

Harshita Srinivas
IIITD
harshita19244@iiitd.ac.in

Hardik Dudeja
IIITD
hardik19422@iiitd.ac.in

Mohit Bhar
IIITD
mohit19256@iiitd.ac.in

Saksham Mehla
IIITD
saksham19270@iiitd.ac.in

## Abstract

*With the growing popularity of e-commerce websites like Amazon, Flipkart, and Myntra, many products available on these platforms have significantly increased. Owing to this, the product classification problem has gained practical significance in the industry. The project aims to address the product classification problem by deeply taking into account natural language processing techniques. It focuses on applying natural language processing to obtain a clean dataset consisting of product descriptions. This has been achieved by eliminating common words, stop words and generic product categories. Additionally, count vectorizers are implemented to convert text data into its appropriate vector representation. Various modern machine learning algorithms are applied to achieve accurate product classification for catalogues consisting of lakhs of products on the obtained vectorized data. In retrospect, this project finds intensive application in the currently booming E-commerce industry.*
*Source code: [GitHub]*

## 1. Introduction

With the growing popularity of e-commerce websites like
Amazon, Flipkart and Myntra, the number of products available on these platforms has greatly increased. Owing to this, the product classification problem has gained practical significance in the industry. A well-built product taxonomy will allow users to easily navigate the website and explore their desired products leading to higher conversion rates and improved user experience. In contrast to manual product classification, where merchants may classify a single product into multiple or incorrect categories, an automated classification system eliminates these inconsistencies. Additionally, appropriate product categorization will also improve search relevance on the web leading to greater traffic being directed towards one's website. Thus, our team was motivated to work on this problem as it aligns with our vision to solve a real-world problem using a diversified dataset and applying a number

of machine learning algorithms coupled with a few natural language processing technique

## 2. Literature Survey

1. Don't Classify, Translate: Multi-Level E-Commerce Product Categorization Via Machine Translation [1] This paper discusses a novel approach to the product classification task. In most proposed methods, building a well-versed product taxonomy looks at the process as a mere classification task. But the taxonomical tree is highly hierarchical with thousands of leaf categories. The approach works on utilizing the above procedure, thereby achieving better results without incurring extra cost. It outputs the existing root to leaf paths in the product taxonomy and also outputs new paths that do not exist in the taxonomy. The product catalogue is represented as a directed acyclic graph(DAG). This helps to visualize the taxonomy more authentically, as psychologically, humans tend to view products differently. The approach takes the input description as a sentence. It then uses a specified number of tokens(m) in the source language and finds its best translation in the target language. It uses novel machine transition models like the Attentional Seq2Seq model and the NMT based transformer model. Using conventional RNNs was time-consuming for the authors' approach; hence novel machine translation algorithms were utilized. Lastly, the above-discussed models were employed to achieve the desired results. The weighted F-score for multi-class predictions was used as an evaluation metric to evaluate the model performance. The authors observed that the model could prune and restructure the taxonomy tree at the appropriate depth to describe the product accurately from the obtained taxonomy trees. By changing the overall mode outlook, the authors depict that machine translation may help product classification more efficiently.

1.2. GoldenBullet: Automated Classification of Product Data in E-commerce [2] This paper provides an insight into a software environment targeted at providing a complete solution to the product classification problem. It discusses B2B marketplaces' present popularity and the need for accurate product descriptions to have a suitable classification model. Different product vendors often use different vocabularies to describe their products. A standardized product description can ease the overall process for the classifier. In addition, it discusses the present United Nations Standard Products and Services Code (UNSPSC) which is widely used to classify products in are hierarchical fashion by assessing specific codes to each product. Golden bullet, the proposed tool developed by the authors, aims to automate every step in product classification, i.e. from information retrieval to product classification with additional UI support to manually input data. The paper discusses a naive approach for product classification.  For the classification task, pre-processed product data is used, and the imported UNSPC dataset is employed. Each product description is treated as a separate query, and the classification task is viewed as an information retrieval task. The authors developed the golden bullet, a tool that uses a diversified training set to achieve the desired results. It uses the vector space model method for information retrieval that finds a mat between the UNSPSC commodities and the product description of the data sample. Then the K-nearest neighbour classifier is implemented to search the most similar class among several categories. Then, a simple naive Bayes classifier was employed to test the model. Golden bullet was implemented as a java-based client-server application. The tool was evaluated on about 40,000 real-world product descriptions. The model achieved an overall accuracy of about 80% across product descriptions in both English and French. This approach is prone to several shortcomings, as modelling and mapping efforts are needed to classify the data. Nevertheless, the application is able to achieve decent accuracy to cater to real-world scenarios. It works at an intersection of natural language processing, information retrieval, machine learning using large volumes of manually classified data.

3. Machine Learning Based Product Classification for eCommerce[3]. This research paper explores classification based on users' search phrases. Here original product names and classes are supplemented with users' search phrases which improved the classification accuracy significantly. The research was conducted in two phases. In the first phase, product classification was done on the basis of names & classes of products. In the second phase, User generated content (search phrases) were used to classify products. At last, the results of both the phases were compared.  The best results of text classification were reported in experiments when linear algorithms have been applied like multiclass logistic regression, decision trees, or support vector machine (SVM) with linear kernel. Accuracy, Precision, Recall, and F1-score were used for quality measurement. It was observed that merging search phrases with original product names improves classification accuracy by almost 50%. To conclude, the use of user-generated content for classification, though causes ambiguous classification, has a positive influence on grouping products in a more user-friendly way.

# 3. Dataset Features

## 3.1 Dataset Description

The dataset for the product classification problem consists of 100 product categories, containing 1000 items corresponding to each category. Each product in the dataset consists of a title, its corresponding description and its brand name. It also comprises of bullet points for each data item that displays product descriptions in a more comprehensive manner. These columns are evaluated together to make accurate product category predictions.

| Feature | Datatype |
|---|---|
| **Title of the product** | String |
| **Description** | String |
| **Brand Name** | String |
| **Important features regarding the product** | Object |

Table 1. Raw Features in dataset

## 3.2 Preprocessing

For data preprocessing, we used an NLP based approach. Firstly, null valued columns were eliminated from the data. Then, each data point was converted into an independent document, a tuple containing the product title, description and bullet points together. Further evaluation was performed on the obtained list of tuples. Each of these tuples was split further by category, i.e. tokenization was performed on the text document. Tokenization enabled us to perform POS tagging and lemmatization on the dataset based on the category. We appropriately classified the words present in each category description as appropriate parts of speech for further evaluation.

## 3.3 Lemmatization

We employed the WordNet lemmatizer to lemmatize categorical data and perform appropriate preprocessing. Every document tuple was cleaned, and stop words and punctuations were eliminated.

### 3.4 Dataset Cleaning

Additional spaces in the product descriptions were removed. After performing initial processing, each word in the tokenized document was processed as follows. In case the word was a stop word or alphanumeric, it was eliminated from the tuple.  In the dataset, products with no defined categories were labeled as 'Generic' products. Since we wish to perform a product classification task on the given model, such instances do not add much value to our dataset. Hence 'Generic' category products were eliminated. Additionally, it was observed from the model evaluation that words of length less than two do not add any value to the description. Hence, all such instances were eliminated. A word was included in the modified document tuple only after it passed all the above checks. Since the data was initially split on the category and description, after preprocessing, the product category was again included in the document tuple to ensure more accessible model training.
Lastly, words that appear in about 95% of the data points were eliminated from the dataset to improve model performance. This is because such words do not help us differentiate between categories or add any overall value to model predictions. This concluded the comprehensive data preprocessing.

### 3.5 Dataset Visualization

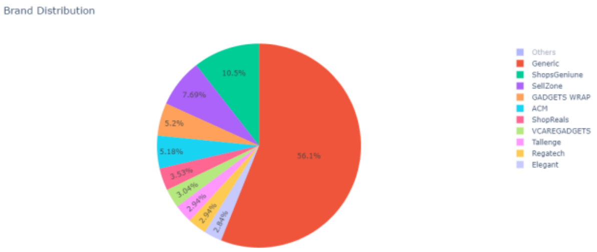


Figure 1:  Product distribution in various categories

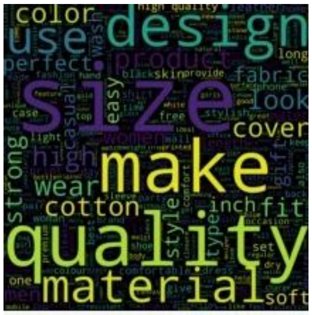


Figure 2:  Word Cloud displaying words with highest occurrence across categories.

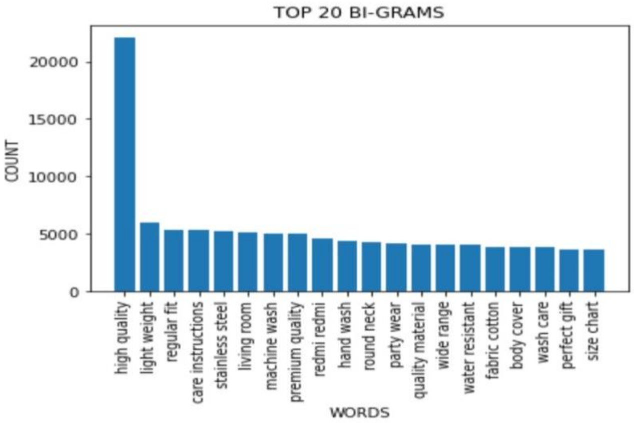


Figure 3: Top 20 bigrams across categories

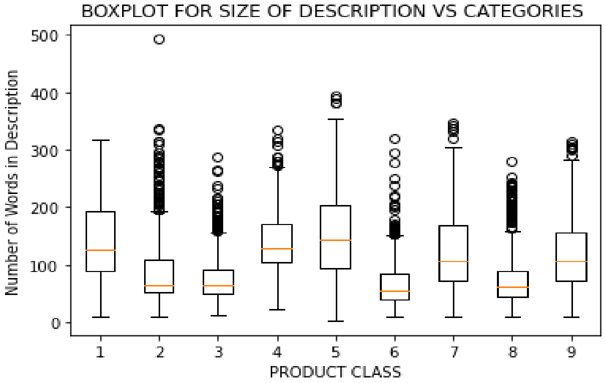


Figure 4: Size of description across categories

## 4. Methodology

The count vectorizer was used to convert text data into its appropriate vector representation based on its frequency. Top 2000 words (in terms of frequency) are considered as the features. On the obtained vectors, we trained suitable classification models.

### 4.1 Logistic Regression

We implemented logistic regression, one of the most commonly used classification techniques. We utilized the implementation provided in the Sklearn library. The parameters that have been adjusted are based on relevance to the classification task. These include penalty, max iterations, C, and the solver to be utilized for the descent. We implemented regression with both L1 and L2 loss functions in order to evaluate the performance holistically. Additionally, we also implemented logistic regression with a stochastic gradient descent solver. This was done to assess the performance against batch gradient descent. In SGD, the tweaked parameters include alpha (regularization parameter), loss, average, eta0, learning rate, maximum iterations. We used Grid Search CV to find the optimal training parameters.

| Feature | Optimal value |
|---|---|
| Penalty | None |
| Max iterations | 500 |

Table 2: Optimal features for plain Logistic regressor

| Feature | Optimal value |
|---|---|
| Penalty | L1 |
| Max iterations | 500 |
| Solver | Liblinear |
| C | 0.1 |

Table 3: Optimal features for Logistic regressor with L1 loss

| Feature | Optimal value |
|---|---|
| Penalty | L2 |
| Max iterations | 500 |
| C | 0.1 |

Table 4: Optimal features for Logistic regressor with L2 loss

| Feature | Optimal value |
|---|---|
| Average | True |
| Max iterations | 500 |
| Alpha | 0.001 |
| Loss | Log |
| Eta | 0.01 |
| Learning rate | constant |

Table 5: Optimal features for SGD regressor

## 4.2 Naive Bayes

Naïve Bayes models provide a simple benchmark for further evaluation of applied models for the classification task. Hence, in order to serve as a benchmark for a large amount of data, we implemented the Naïve Bayes classifier. Naive Bayes classifier assumes that the effect of a particular feature in a class and does not consider other features. Hence it serves useful for classification tasks to determine the class to which a product may belong. We employed two naïve bayes classifiers for benchmarking.

### 1. Multinomial Naive Bayes

This probabilistic learning method is most commonly used alongside natural language processing techniques. It uses a multinomial distribution for the evaluation of the posterior probability(theta) using relative frequency estimates.

$$\hat{\theta}_{yi} = \frac{N_{yi} + \alpha}{N_y + \alpha n}$$

$N_y$ represents the total number of features and $N_{yi}$ represents the count of each feature. α is a smoothing Laplace smoothing parameter to prevent instances of zero probability. In our implementation, we tweaked the alpha value as per requirement.

| Feature | Optimal value |
|---|---|
| Alpha | 5 |

Table 6: Optimal features for Multinomial Naive Bayes

### 2. Bernoulli Naive Bayes

Bernoulli naïve is useful for discrete data and models the presence or absence of a feature in a tuple. Hence, the features are evaluated as true or false values, which serve useful for classification purposes. For instance, in the product classification task, Bernoulli naïve Bayes helped us model the features associated with a particular product category and its presence or absence. The following decision rule is applicable for Bernoulli naïve bayes.

$$P(x_i \mid y) = P(i \mid y)x_i + (1 - P(i \mid y))(1 - x_i)$$

It penalizes the non-occurrence of a feature *i* which may be an indicator for a cateory y.

| Feature | Optimal value |
|---|---|
| Alpha | 0.01 |
| Binarize | 0.5 |

Table 7: Optimal features for Bernoulli Naive Bayes

## 4.3 Decision Trees

Decision tree model is a powerful classifier which utilizes entropy or gini index as a criterion to decide feature importance. In our implementation, we adjusted features like maximum depth of the tree, minimum number of samples required to split an internal node and the minimum number of samples required to be at a leaf node to understand their influence of the model. We used Grid Search CV to find the optimal training parameters.

| Feature | Optimal value |
|---|---|
| Max_depth | 100 |
| Min_sample_leaf | 1 |
| Min_samples_split | 5 |

Table 8: Optimal features for Decision Tree

## 4.4 Random Forest

Random forests have a lot of hyperparameters which can be tweaked. The ones that we adjusted in our implementation include the number of estimators, max depth of the trees and minimum number of samples required to split an internal node and the minimum number of samples required to be at a leaf node. To find the optimal values of these parameters, we used validation curves for each of the above-mentioned parameters.

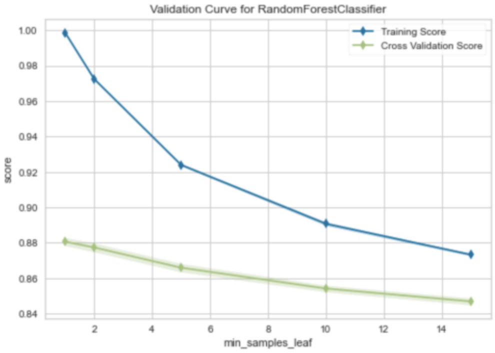
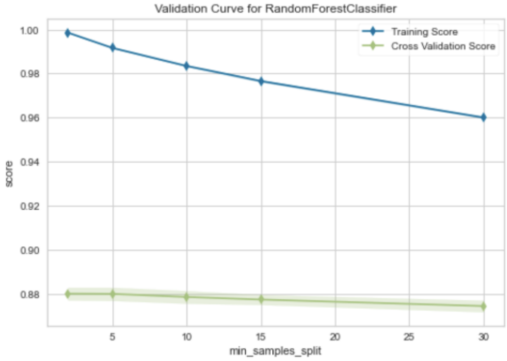
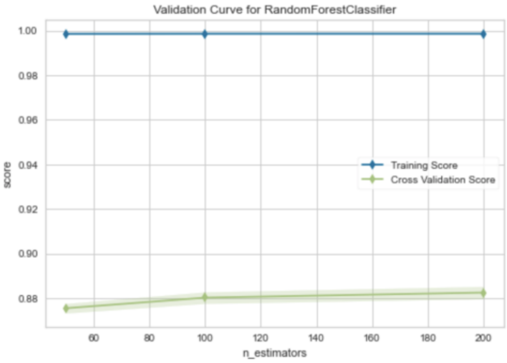


Figure 5: Validation curve for Random Forest Classifier at min_samples_leaf



. Figure 6: Validation curve for Random Forest Classifier at min_samples_split



. Figure 6: Validation curve for Random Forest Classifier at n_estimators

| Feature | Optimal value |
|---|---|
| Max_depth | 100 |
| Min_sample_leaf | 2 |
| N_estimators | 200 |
| Min_samples_split | 5 |

Table 9: Optimal features for Random Forest

## 4.5 Multilayer Perceptron

Multiplayer perceptrons with multiple hidden layers serve useful for classification tasks. We employed an MLP on the vectorized dataset using different activation such as Leaky-ReLU, ReLU, Tanh, Linear, Sigmoid.

| Feature | Optimal value |
|---|---|
| hidden_layer_sizes | [64, 32] |
| Max iterations | 100 |
| activation | logistic |

Table 10: Optimal features for MLP

## 4.6 K-Nearest Neighbors

K-NN is the most widely used classification algorithm for multiclass classification. The advantage over other algorithms is that points are classified based on their relative distance from average centroid. Each product vector was represented in an N-Dimensional space according to our features. Then, we evaluated the possible products in that category using the following K products. This was done based on the Euclidean distance measured from the average centroid.

| Feature | Optimal value |
|---|---|
| hidden_layer_sizes | [64, 32] |
| Max iterations | 100 |
| activation | logistic |

Table 11: Optimal features for K-NN

## 4.7 SVM

Support vector machines are very effective in high dimensional spaces. SVC and implements the "one-versus-one" approach for multi-class classification. A total of n_classes * (n_classes - 1) / 2 classifiers are constructed. We used SVM classifier at default parameters to determine the optimal product category.

|  | Training Data |  |  |  | Testing Data |  |  |  |
|---|---|---|---|---|---|---|---|---|
| **Model** | A | P | R | F1 | A | P | R | F1 |
| **Logistic Regression (Loss: L2)** | 0.91 | 0.91 | 0.90 | 0.90 | 0.87 | 0.87 | 0.87 | 0.87 |
| **Logistic Regression (Loss: L1)** | 0.89 | 0.89 | 0.89 | 0.89 | 0.86 | 0.86 | 0.86 | 0.86 |
| **Logistic Regression (Loss: None)** | 1.00 | 1.00 | 1.00 | 1.00 | 0.84 | 0.84 | 0.84 | 0.84 |
| **SGD Classifier** | 0.89 | 0.89 | 0.89 | 0.89 | 0.86 | 0.86 | 0.86 | 0.86 |
| **Multinomial Naïve Bayes** | 0.82 | 0.83 | 0.82 | 0.82 | 0.80 | 0.80 | 0.80 | 0.79 |
| **Bernoulli Naïve Bayes** | 0.79 | 0.81 | 0.79 | 0.80 | 0.75 | 0.78 | 0.75 | 0.76 |
| **Decision Tree** | 0.95 | 0.97 | 0.97 | 0.97 | 0.81 | 0.82 | 0.81 | 0.82 |
| **Random Forest** | 0.96 | 0.97 | 0.97 | 0.97 | 0.88 | 0.88 | 0.88 | 0.88 |
| **MLP** | 1.00 | 1.00 | 1.00 | 1.00 | 0.86 | 0.86 | 0.86 | 0.86 |
| **K-NN** | 0.86 | 0.85 | 0.85 | 0.85 | 0.79 | 0.76 | 0.76 | 0.77 |
| **SVM** | 0.92 | 0.92 | 0.92 | 0.92 | 0.85 | 0.85 | 0.84 | 0.84 |

## 5. Results and Analysis

We analyzed each of the considered models based on various performance metrics such as accuracy, precision, recall and F1 score (as seen in the table above). In a few instances, our model performs exceptionally well on the train data; however, its comparative performance in the testing data is not up to the mark. In such selected instances, such as the logistic regressor with the L1 loss, we observe that there may be an instance of model overfitting. From the observed models it is evident that the SGD classifier has the best performance overall owing to low variance.

## 6. Conclusion

Through this project, we applied various machine learning models, which helped us obtain a clear understanding of concepts. We discovered how different models perform better depending on the type of data and tackled various challenges like over-fitting and under-fitting of models. Additionally, we gained a deep insight into natural language processing techniques. We utilized word vectorizers and lemmatizers which helped us understand the optimal procedure towards applying NLP. We were able to provide a viable solution to the product classification task in terms of a model that achieved high accuracy alongside decent recall and precision scores. This was on account of well-applied natural language processing techniques. We learnt that product descriptions provide the maximum information alongside the brand name in the product classification task. Product names also have a significant impact. However, ambiguous naming can often serve as an outlier, hampering the classification task. The metric scores are highly promising and show a deep resemblance to the previous work done in the field. Future improvements on the approach can be made by using more diversified datasets combined with deep learning mechanisms to yield useful results.

## 7. References

[1] Ding, Y., Korotkiy, M., Omelayenko, B., Kartseva, V., Zykov, V., Klein, M., ... & Fensel, D. (2002, April). Goldenbullet: Automated classification of product data in e-commerce. In *Proceedings of the 5th international conference on business information systems*.

[2] Li, M. Y., Kok, S., & Tan, L. (2018). Don't Classify, Translate: Multi-Level E-Commerce Product Categorization Via Machine Translation. *arXiv preprint arXiv:1812.05774*.

[3] Amazon ML Challenge: [https://www.hackerearth.com/challenges/competitive/amazon-ml-challenge/]

[4] Mieczysław Pawłowski (2021) Machine Learning Based Product Classification for eCommerce, Journal of Computer Information Systems, DOI: 10.1080/08874417.2021.1910880