

Steps

1. Data Pre-processing

- (a) I downloaded the given spreadsheet in a csv format. It had 15 columns. I used the pandas library read_csv function to read the csv file and store it in the messages dataframe.
- (b) As we only need to predict major product categories from the description, the only relevant columns would be 'product_category_tree' and 'description'. Thus I created a product dataframe which stores these 2 columns. I also remove rows with null values.
- (c) The product category tree lists the subcategory of the product too, but as we need only the primary category, I create a new column in my products dataframe 'primary_category' and store it after extracting it from the product_category_tree. I do that by partitioning the string based on '>>' and then selecting the first partition and removing the first 2 chars i.e. '['
- (d) We now have the product description and its primary category.

2. Data Visualisation and exclusion of categories

- (a) I use the describe function and notice that there are 266 primary categories spanning 19998 products. The most frequent is 'Clothing' with 6197 products.
- (b) Seeing that clothing alone spans about 30% of the dataset, I get an intuition that maybe it's only some of the major categories with sufficient data for an accurate analysis.
- (c) So I calculate the cumulative frequencies and thus cumulative % of the top 28 primary categories .
- (d) These top 28 most frequent categories alone occupy 98.3% of the dataset i.e. 19661/19998 items. Thus we can remove all the remaining categories for a more accurate data analysis from the set.
- (e) The remaining 337 products are spread over 234 categories, and there are <10 entries for each such category, thus we should exclude these categories.

3. Data Cleaning

- (a) I clean the data obtained- i.e. remove punctuation marks, extra spaces, special characters etc. i.e. everything except for regular expressions having [a-zA-Z] is removed.
- (b) Next, I remove the stop words in the english language like "the, a , and" etc that don't help much in distinguishing one category from another.
- (c) This is followed by the stemming of the words in the description, to get their roots.
- (d) The data has finally been cleaned

4. Data Training and Testing

- (a) First of all I split the entire data set into 2- the training data (80%) and testing data(20%).
- (b) I train the model to predict the category using the training data and test its accuracy using the testing data(20%).
- (c) The model training is done using the count vectorizer, which uses the bag of words technique to find the frequency of occurrence of several words across descriptions.
- (d) I use the Multinomial Naive Bytes Algorithm and provide it with the countvectorizer obtained and the associated descriptions to train the model.

- (e) I use the trained model so obtained to predict the categories of the remaining 20% of the dataset and the accuracy obtained is **93.77%**

Notes and Observations:

1. I used the count vectorizer instead of the tf_idf vectorizer, as the classes in the dataset are skewed, as discussed in the visualization section. Tf_idf is better to use when the data is almost equally distributed among all classes. I tried using tf_idf vectorizer, and obtained a reduced accuracy of 86.33%
2. We can try lemmatization instead of stemming the words:
Using that I get an accuracy of 93.69%, which hardly differs from the accuracy obtained using stemming.