Product Categorization:

Brief summary of implemented solution:

1) The text from the “description” field of product_data.json was used
2) Text preprocessing: Removal special characters, numbers, html tags, lowercasing of all text etc,.
3) Lemmatization and stop-words removal. Domain specific stop-words were added to remove words related to shipping, size, gender, brand name, fabric maintenance etc,.
4) If description was empty after preprocessing, categorize the product as “Others”
5) Tf-idf vector generation of “description” field and hierarchical k-means clustering using cosine distance was used to obtain product categories.
6) Most frequent words in each cluster was used to name product category. Some of the clusters were subject to second round of clustering depending on the results.
7) Sometimes most frequent words of cluster were useful in feedback to include terms in domain specific stopwords
8) Additional categories were found apart from the categories (ex: pant) mentioned in the product_categories.txt and some categories were not identified (ex: Rompers)

Why are you designing the solution in this way?

a) Given unlabelled data, and the manual effort involved in label generation were primary motivators for unsupervised solution implemented.
b) Additionally, unsupervised methods provide information on inherent grouping among the given product category descriptions and may lead to discovering new categories

What are the aspects that you considered when designing?

a) Choice between supervised and unsupervised
b) Choice between count vectorizer (Tf-idf, count) vs embedding (doc2vec, word2vec etc). Averaging/aggregating vector embeddings of a description from its words would not be a differentiating indicator. A brief experiment on doc2vec did not yield initial clustering quality as of count vectorizers.
c) Choice between K-means, DBSCAN, LDA for clustering. K-means with cosine similarity was chosen as initial exploratory method, due to the simplicity of model

d) Choice of Stemming, Lemmatization or both
e) How to devise domain specific stop-words

What are the cases your solution covers, how are they covered and why are they important?

a) Solution uses “description” text field as it provides information on the product. Only non-empty description fields were considered for clustering
b) Solution assumes the “text” is in English for text-preprocessing by enabling language specific libraries, as the majority of product descriptions were in English

What are the cases your solution does not cover and what are the ways you can extend your current solution for them?

a) “Url text” of a product could be useful in addition to “description”. As the given product_json is a collection of products from multiple sources (jcrew, urban outfitters, shopify, gap, etc.). Each source’s url needs separate text-processing to extract useful text due to variance in their url descriptions.
b) Images are not used in the initial method to aid in categorization. Some urls were not valid. An ML approach to include heterogeneous information of both product text and product image would be helpful. However, extraction of useful parts of image (i.e if description talks about a t-shirt and the model in the image shows both pant and t-shirt), and correlation between image results and product url is required to obtain final categorization.
c) Non-english description were not translated. Separating non-english test cases or using language translators might have been useful
d) Devising domain specific stop-words is critical to design, current set is based on the data available and might not be robust across various test cases. Named entity recognition based text processing could be used to devise stop-words