MIDAS Evaluation Task 3 : Build a Model to Predict Product Category using Description

About Me

Name : Aryan Gupta
Email : aryangupta973@gmail.com
Github : https://github.com/withoutwaxaryan
LinkedIn : https://www.linkedin.com/in/wwaryangupta/

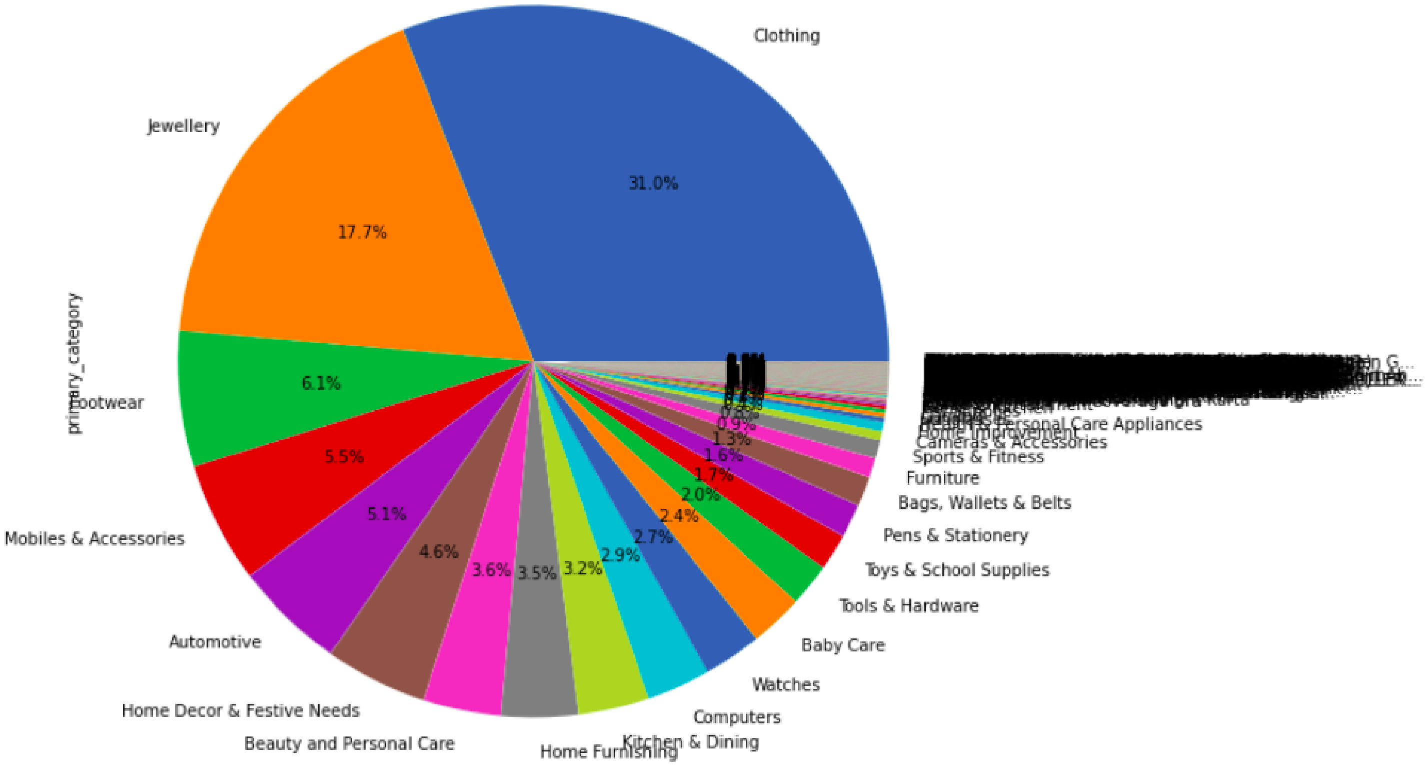
Included Files

- Dataset - flipkart_inventory.csv
- Prediction Model - prediction_model.ipynb
- Experiment Log - experiment_log.ipynb
- Requirements.txt

Approach

I approached the task as a Supervised Text Classification problem.

1. **Data Exploration**
    - Dropped columns not relevant to the task
    - Created a column ‘primary_category’ after splitting ‘product_category_tree’ to get the root category.
    - Explored primary_category and it's unique rows using value_counts & bar plots.
    - Made changes to split the product_root_category based on the given data. (Experiment #1 in Experiment Log).
    - Came across certain outliers which may or may not harm the model - kept them for later processing.



Data Distribution of categories

2. Setup Machine Learning Models

- Created Training & Test Datasets (7:3) with input as ‘description’ and output as ‘primary_category’.
- Comparison of Text Feature Extraction with CountVectorizer & TFIDFVectorizer - TFIDF gave better accuracy. For the following ML Algorithms, TF-IDF was used with the respective ML Pipeline.
- Tried out a number of Supervised ML Algorithms :
  - Linear SVC - Accuracy → 96.33 %
  - Naive Bayes - Accuracy → 78.16 %
  - Logistic Regression - Accuracy → 93.86 %
  - K Nearest Neighbours Classifier (KNN) - Accuracy →93.82 %
  - Random Forest Algorithm - Accuracy → 92.82 %

- The Accuracy of the Model was measured using Sklearn's Classification Metrics :
    - Overall Accuracy Score
    - Confusion Matrix
    - Classification Report (Precision, Recall, F-1 Score).

The highest accuracy was achieved by Linear SVC, using TF-IDF Vectorizer.

3. **Data Processing** (Done iteratively to see the effects through the Model's accuracy)
    - Data preprocessing techniques used for Description column :
        - Lowercase
        - Removing Links, punctuations, codes (eg. VUX342)
        - Removing single alphabets and extra spaces
        - Removing stopwords supplied by NLTK corpus
    - Sample run of the model (Linear SVC pipeline with TF-IDFVectorizer) showed an improvement of 0.12 % of the model. Accuracy - 96.45 %
    - Experimented to improve accuracy by using additional stopwords from the given dataset. However, it decreased accuracy by 0.14%. (Experiment #2 in Experimental Log).
    - Removed those outliers (primary categories) which had 1 to 2 items only. This helped improve the quality of the dataset & hence the accuracy of the model to 97.6 % (increase of 1.15 %).
    - Finally, I manually curated ~ 20 rows by classifying them into better primary categories. This also involved curating a category of 'Sunglasses' to a better super category of 'Eyewear'. This helped improve the accuracy of the model to 97.92 % (increase of 0.32 %).

My reason for manual curation, and not dropping the rows, was a particular category of 'Household Supplies' which although only had 4 items, still was an important primary category to be considered. Had I dropped all categories consisting of less than 10 items, this category would have been deleted too. Also, since each of these 20 manually curated categories had 3 items or more, I was able to add 60 + correct entries and improved the dataset, with little time and effort.

Conclusion

The best model turned out to be Linear SVC with an accuracy of 97.92 %.

I think the accuracy of the Model can be increased in the following ways :

- Some of the chosen Primary Categories are similar i.e. Home Decor, Home Furnishings, Home Entertainment. This may be used to create a single category known as 'Home'. Creating a superclass will always improve accuracy.
- Other columns such as 'product_name' and 'product_specifications' could have been used as input features to improve the model.
- Using NLP based Deep Learning Models involving Transformer Models, and NLP techniques such as Bag of Words, and Word2Vec, Word Embeddings using Gensim etc.

References

1. Similar Kaggle Dataset involving News Category (Link)
2. Beginners Guide to Data Cleaning & Feature Extraction (Link)
3. Text Analytics Datacamp (Link)
4. Classification Models with Sklearn (Link)
5. Lots of StackOverflow!