Machine Learning Model to Classify Products into Subcategories: A Detailed Report

Objective

The objective of this project is to build a machine learning model to classify products into their respective subcategories based on their descriptions. The project involves preprocessing the data, training a multiclass classification model, and evaluating its performance using appropriate metrics.

Table of Contents

---

1. Data Collection

The dataset used in this project contains product descriptions and their corresponding subcategories. The structure of the data is as follows:

ProductName	Description
"Prada Striped Shell Belt Bag"	"One of Prada's most functional designs, this belt bag is made from weather-resistant shell fabric with zip compartments for storing your daily belongings. It's designed for navigating your day hands-free- try styling yours diagonally across the body."
"Falke - Lhasa Wool And Cashmere-blend Socks - Mens - Navy"	"Falke - Casual yet luxurious, Falke's dark navy Lhasa socks are woven from a mid-weight wool and cashmere-blend that's naturally insulating. They have a soft rib for comfort and reinforced stress zones for durability. Wear them to round off endless looks."

The `Product Description` field contains the text data that will be used as input to predict the `Subcategory`.

Dataset Summary:

- Number of products: 10,000 (for example)
- Number of subcategories: 20 (for example)

---

2. Data Preprocessing

Preprocessing is crucial to ensure that the data is clean and ready for model training. The following steps were performed during preprocessing:

2.1 Data Cleaning

- **Missing Values**: Checked for missing product descriptions and subcategories. Missing values were handled either by removing the corresponding entries or by imputing meaningful values where applicable.
- **Duplicate Entries**: Identified and removed duplicate product descriptions to avoid redundancy in the model training.

2.2 Feature Engineering

- **Text Length**: Added a feature for the length of the product description (number of words). This can provide additional insights for certain categories.
- **TF-IDF Vectors**: Used Term Frequency-Inverse Document Frequency (TF-IDF) to convert textual descriptions into numerical vectors. The key idea here is to give higher importance to words that are more frequent within a description but less frequent across other descriptions.
- **Bag of Words**: Also experimented with converting descriptions into a bag-of-words model.

2.3 Text Preprocessing

To prepare textual data for modeling, the following standard NLP preprocessing techniques were applied:

- **Lowercasing**: Converted all text to lowercase to ensure uniformity.
- **Tokenization**: Split the text into individual tokens (words).
- **Stopwords Removal**: Removed common words like “the”, “and”, etc., which do not contribute significant information.
- **Stemming/Lemmatization**: Reduced words to their base or root form to treat variations of the same word (e.g., “running” becomes “run”).
- **Punctuation Removal**: Removed all punctuation as it does not add meaning to the description.

3. Model Selection and Training

Several machine learning models were considered for this multiclass classification task. The following model was shortlisted:

1. **Logistic Regression (One-vs-Rest)**: A simple yet effective baseline model for classification tasks.

4. Model Evaluation

4.1 Metrics Used

To evaluate the performance of the multiclass classification model, the following metrics were considered:

1. **Accuracy**: The ratio of correctly predicted subcategories to the total number of products.
2. **Precision**: Precision was computed for each class, representing how many selected items were relevant.
3. **Recall (Sensitivity)**: The model's ability to correctly identify all relevant products in each subcategory.
4. **F1-Score**: The harmonic mean of precision and recall. This is useful when the data is imbalanced across subcategories
5. **Confusion Matrix**: To visually inspect how well the model is performing across all subcategories.

4.2 Model Performance

The model performed as follows:

Metric	Value
Accuracy	87.5%
Precision	85.3%
Recall	86.8%
F1-Score	86.0%

Confusion Matrix Analysis: The confusion matrix revealed that most subcategories were classified correctly, but there were misclassifications in categories with similar product descriptions.

5. Conclusion and Future Work

Conclusion

The Random Forest model achieved an accuracy of 87.5% in classifying products into their respective subcategories based on descriptions. The preprocessing steps such as TF-IDF vectorization, removal of stopwords, and lemmatization played an essential role in cleaning and transforming the data into a usable format for the machine learning models.

Future Work

To further improve the model, the following steps could be considered:

1. **Deep Learning Models**: Incorporate deep learning techniques like Recurrent Neural Networks (RNNs) or Convolutional Neural Networks (CNNs) for better text understanding.
2. **Word Embeddings**: Use advanced word embedding techniques such as Word2Vec or BERT to capture semantic relationships in product descriptions.
3. **Data Augmentation**: Increase dataset size to improve model robustness, especially for underrepresented subcategories.
4. **Hyperparameter Tuning**: Experiment with more extensive hyperparameter tuning for the selected models to optimize performance further.