Classify new products into an existing taxonomy of product categories for H&M

This capstone project aims to build supervised classification models to predict product categories based on product titles, descriptions and product images.

About the data

This dataset comes from H&M and contains their product metadata and product images

The dataset has 105k rows with 24 columns. Each product has 1 image.

Involves NLP and computer vision techniques

Data Cleaning and Wrangling

- Convert data types into appropriate ones for machine learning
- Identifying which are the relevant features in predicting product category
- Identify missing data and drop null values (every product needs to have both text and image data)

Product Variants - color:

Keep the color variants so the image classification model has more data to train on, but don't include color as a feature.

Keeping these product variants means train/test split needs to be done in a way so these variants don’t leak data.

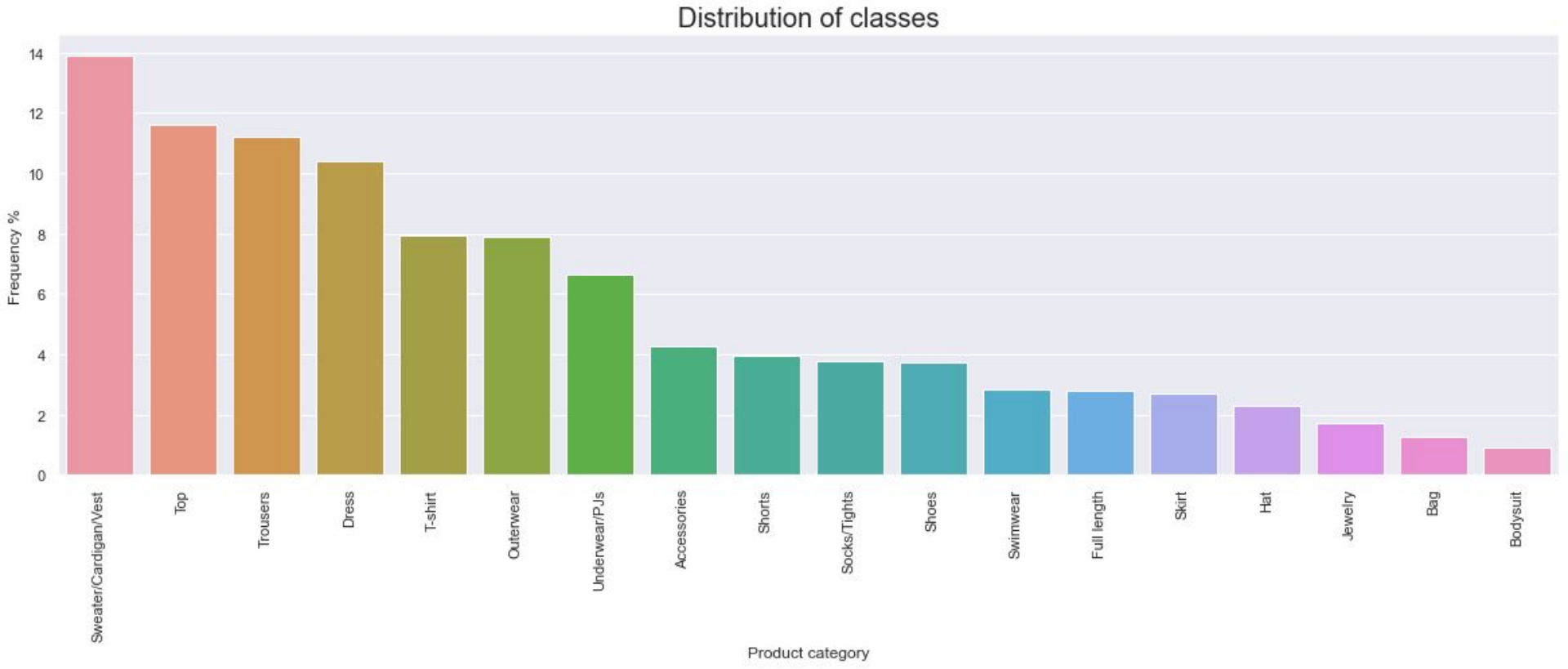
Select the major product categories for prediction

There are 20 product categories

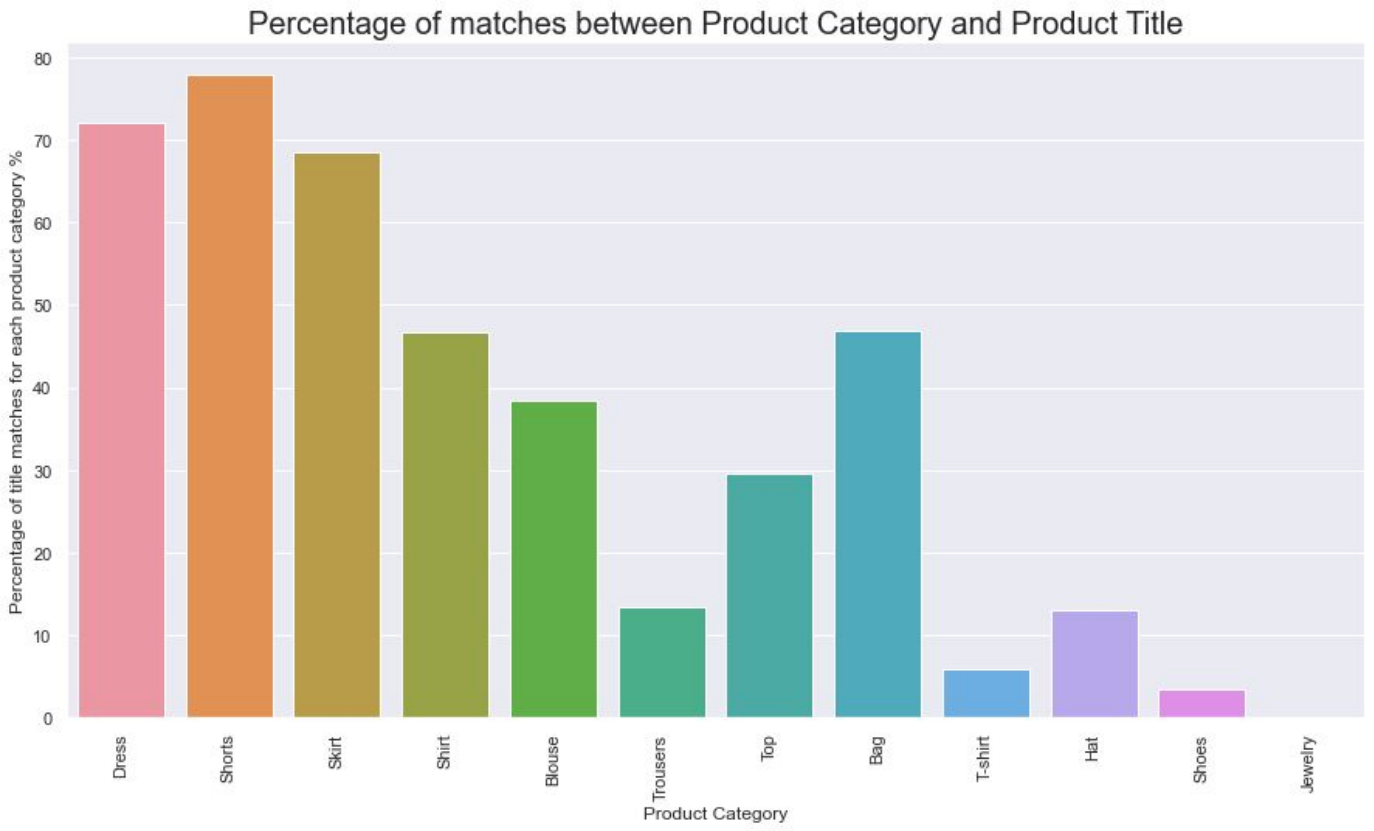
- Dress
- Sweater/Cardigan /Vest
- Trousers
- Outerwear
- Accessories
- Underwear/PJs
- T-shirt
- Blouse
- Shoes
- Top
- Full length
- Skirt
- Shorts
- Shirt
- Jewelry
- Swimwear
- Hat
- Socks/Tights
- Bag
- Bodysuit

Exploratory Data Analysis

Visualize the product categories



Product title vs. Product category overlap

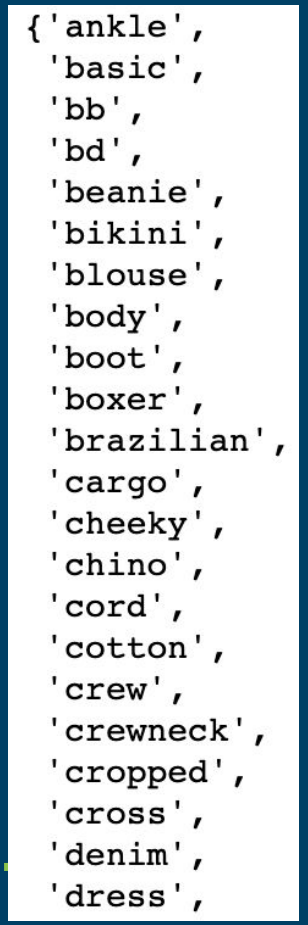


Compare the number of occurrences of the stemmed product title and product category, plot the non zero values as a percentage.

Preprocessing and feature engineering

Create a bag of words from the product title and/or product description, excluding stop words, numbers and punctuation using NLTK.

Choose 100 of the most frequently appearing words as binary features



Create image embeddings

ResNet-34 with pretrained model weights

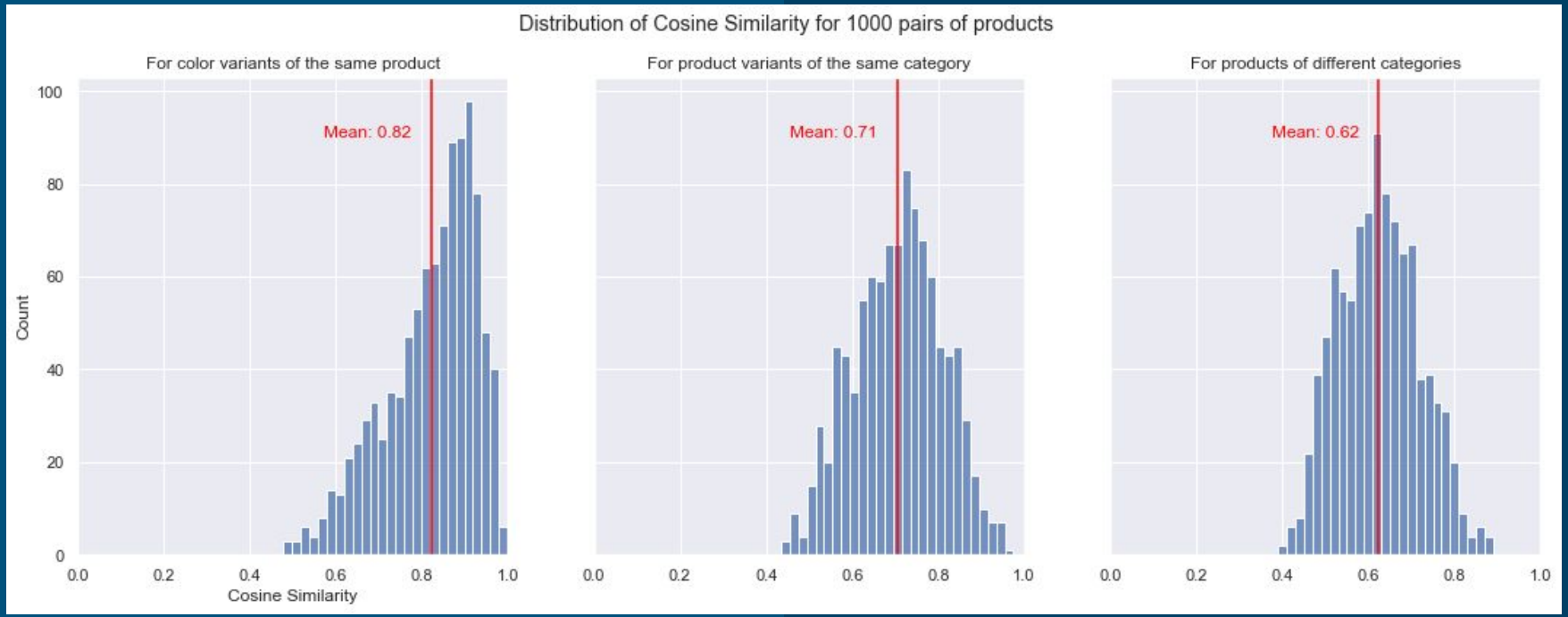
Image preprocessing steps:

1. Resize the image to a square measuring 256 pixels by 256 pixels
2. CenterCrop to 224 pixels
3. Convert image to a Tensor
4. Normalize the tensor image with mean and standard deviation
5. Convert to RGB

The image embedding output from ResNet-34 is 512 columns of features.

Look at similarities between 1000 pairs of image embeddings to check the validity of the embeddings

- Group 1: Color variants should have the most similar image embeddings.
- Group 2: Products in the same product category
- Group 3: Products in different categories should have the least similarity between image embeddings.



Relabel product categories that are incorrect

After training the Linear model, an inspection of prediction errors shows there is a data issue: incorrect product category labels

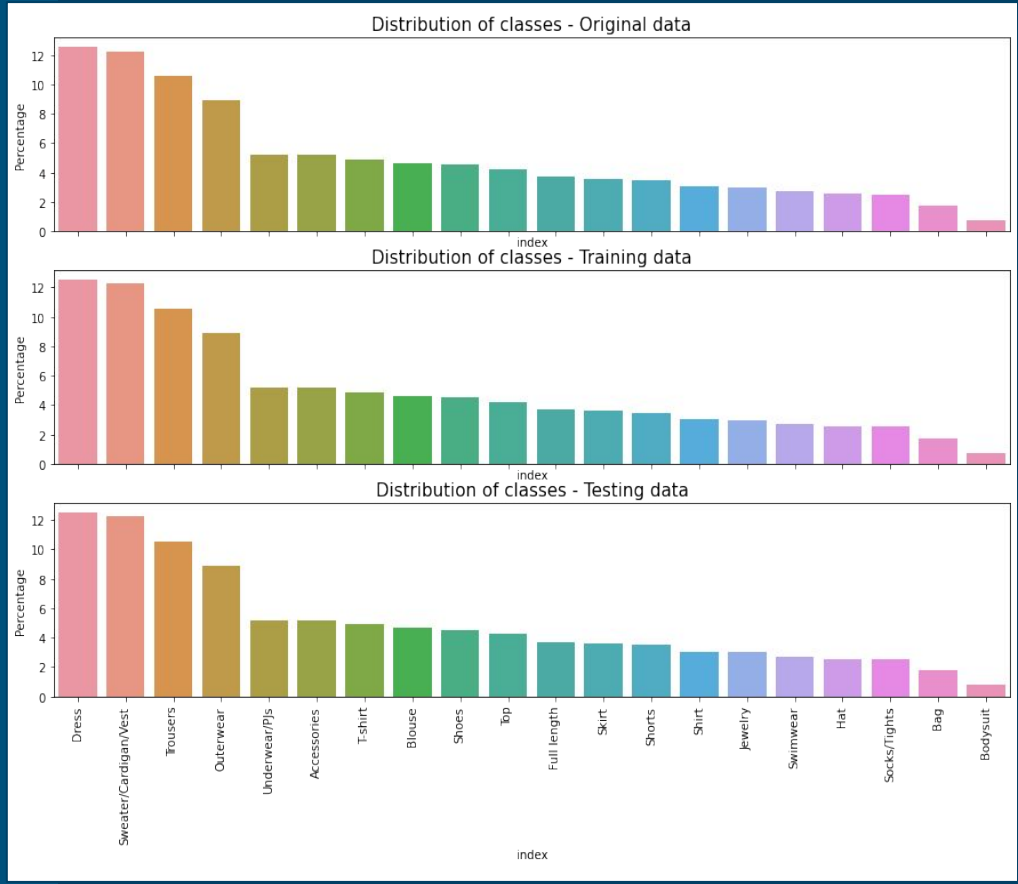
- Focus on the Linear predictions where the model was most confident but predicted incorrectly
- Look at the decision function score, sorted in descending order
- Verify with the product title, description and image to confirm the y classification label was incorrect
- A trend observed in this dataset is that `Shorts` are often labeled as `Trousers`

Group Shuffle Split

70/30 train test split
The data frame has 99332 rows of products.

Use SKLearn's GroupShuffleSplit to group by product variants, otherwise data leakage will occur.

For each product category, perform the split to maintain the same class imbalances and concatenate together to create X_train and X_test.



Machine Learning Models

Baseline Model

Linear model using Elastic Net

Random Forest Classifier

KNN Classifier

Deep Neural Net using Keras

Weighted Average F1-score

Since this is a multi-class classification, the F1 score is calculated for each class in a One-vs-Rest (OvR) approach.

Calculate the weighted average F1-score by taking the mean of all per-class F1 scores while considering each class's support.

Different combinations of features were evaluated on the SDGC model:

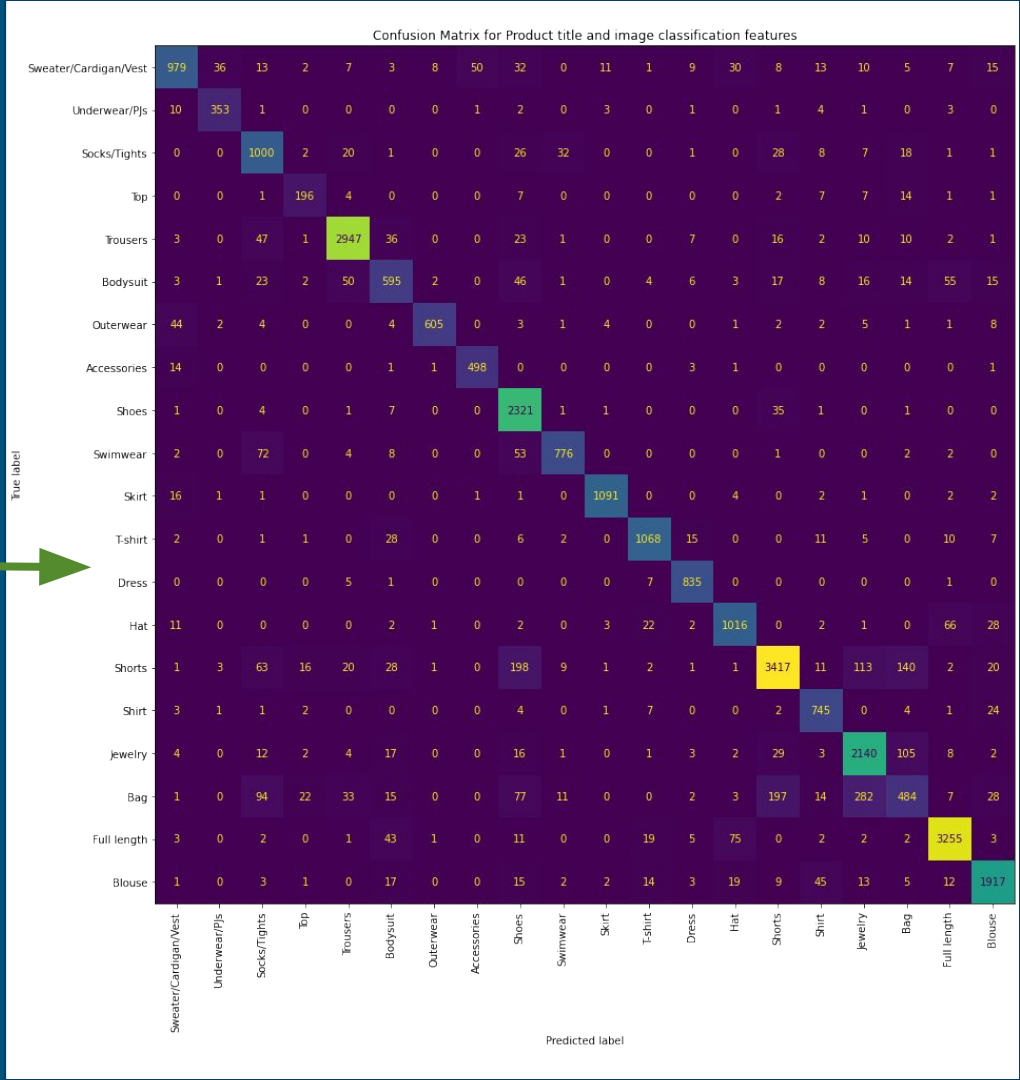
Image embedding features

Product title features

Product title + description features

Product title + image embeddings

Product title + description features + image embedding features



Deep Neural Net using Keras

Steps involved in tuning the neural net model include:

- Shuffling the training and test sets so the y labels (product categories) do not appear in clusters
- One hot encoding the y labels
- This neural net model consists of a single hidden layer of 684 nodes, 20 output nodes
- A dropout rate of 0.25 is chosen to minimize co-adaptation and help reduce model overfitting.
- 20% validation set with early stopping

Conclusion

Overall, the Linear and Neural Net models performed similarly and well enough for the large classes. There is some confusion between categories that are similar or subsets of each other (Accessories and Jewelry, Tops and T-shirt) which is understandable.

Applications of the product category prediction model

- Managing product data by automating labeling of new products
- Maintaining correctly labeled data: these models could flag incorrectly labeled products in an automated way that can save time and money from having to manually process the data.
- Verifying the existing taxonomy makes sense. Having correctly classified products helps customers find what they are looking for and boost sales.

Thank you!

Github portfolio:
https://github.com/melissavhan/H_and_M_product_category_classification