Springboard - DSC

Capstone Project 3 Final Report

Predicting H & M product categories

Melissa Han

February 2023

Table of Contents

List of Figures

List of Tables

1 Introduction - the problem statement

The business problem is classifying new products into an existing taxonomy of product categories for the company H&M. We want to build machine learning models that will help to classify new products and check that product labels are correct.

This capstone project aims to

- Explore the H&M dataset and gain insights into which types of product information would be suitable for feature engineering.
- Build several supervised classification models to predict product categories based on product titles, product descriptions and product images.

2 About the data

2.1 H&M Data

H&M, like many other large ecommerce retailers, face the issue of categorizing inventory into their own existing taxonomy, and maintain the accuracy of their inventory. H&M sell their own branded merchandise manufactured in house, and also sell inventory by other brands/collaborators.

The dataset is from the H&M kaggle competition, which was to provide product recommendations based on previous purchase history of individual customers. Part of the competition's data is a dataset of H&M product metadata and product images. The product categories are provided for each product.

The dataset has 105k rows, where each row is a product. There are 24 columns in the dataset. I like this dataset because it would use both NLP and computer vision. The constraints of this dataset is that there are several product categories that could be used for classification, some more relevant to certain business use cases than others, so determining which one to use is open ended (and might evolve into using a combination of product categories).

3 Data Cleaning and Wrangling

The features at the beginning of the data wrangling stage are:

- Convert data types into appropriate ones for machine learning
- Identifying which are the relevant features in predicting product category
- Identify missing data
- Compare the number of images vs. the number of products to ensure each product has text and image data
- Drop null values

Product category prediction training and testing data follows the following assumptions:

- The product category chosen is the most obvious and correct one, even if an item can fit into several product categories. E.g. A beanie could be labeled as Hat or Accessories, but the most obvious category would be Hat.

To make predictions, the product image and product title must not be missing. The product description is optional.

3.1 Product variants - Color

A product can come in different color variants, leading to many near duplicate products. Instead of removing these color variants, keep them in the dataset so the image classification model has more data to learn from, but don't include color as a feature.

A difference in color does not add any new information about the product to product category classification. The column 'product_code' is the same for all color variants of the same product.

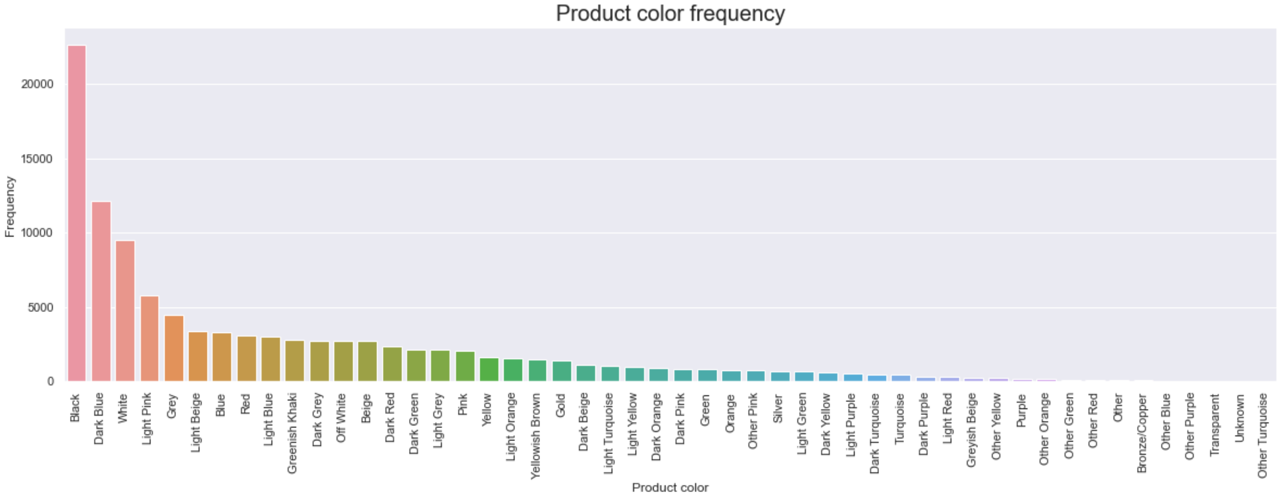


Figure 3.1 Bar graph showing the distribution of color among products

3.2 Select the major product categories for prediction

The target variable is the product categories column 'product_type_name'. Select categories that had a minimum of 200 products.

Then manually combine several of the categories that should belong together:

New category name	Existing product categories
Accessories	Other accessories, Hair/alice band, Gloves, Belt, Sunglasses, Scarf
Jewelry	Necklace, Earring
Hat	Hat/beanie, Hat/brim, Cap/peaked
Shoes	Other shoe, Sandals, Boots, Sneakers
Swimwear	Swimwear bottom, Swimsuit, Bikini top
Full length	Jumpsuit/Playsuit, Dungarees, Garment Set
Underwear/PJs	Underwear bottom, Underwear Tights, Bra, Pyjama set
Outerwear	Jacket, Hoodie, Blazer, Coat
Sweater/Cardigan/Vest	Sweater, Cardigan, Vest top
Socks/Tights	Socks, Tights, Leggings/Tights

Table 3.2 Summary of product category groups created from existing product categories

There are a total of 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

3.3 Relabel product categories that are incorrect

During the modeling stage it was clear looking at the prediction errors that some product categories are mislabeled in the original dataset.
To fix this, focus on the rows where the Linear model was most confident in its prediction but the prediction was incorrect (by sorting the decision function score in descending order). Inspecting the product title, description and image confirmed the original y classification label was incorrect.

A trend observed in this dataset is that `Shorts` are often labeled as `Trousers` in the dataset even though the product title contains the word shorts.

4 Exploratory Data Analysis

4.1 Visualize the class imbalance

The most frequently occuring product category ‘Sweater/Cardigan/Vest’ makes up 14% of the product categories, which means that guessing the category ‘Sweater/Cardigan/Vest’ should be correct 14% of the time.

The smaller classes such as ‘Bag’ and ‘Bodysuit’ make up a much smaller fraction of the product categories and machine learning models may need tuning or be assigned weights so the model isn’t too biased from these smaller product categories.

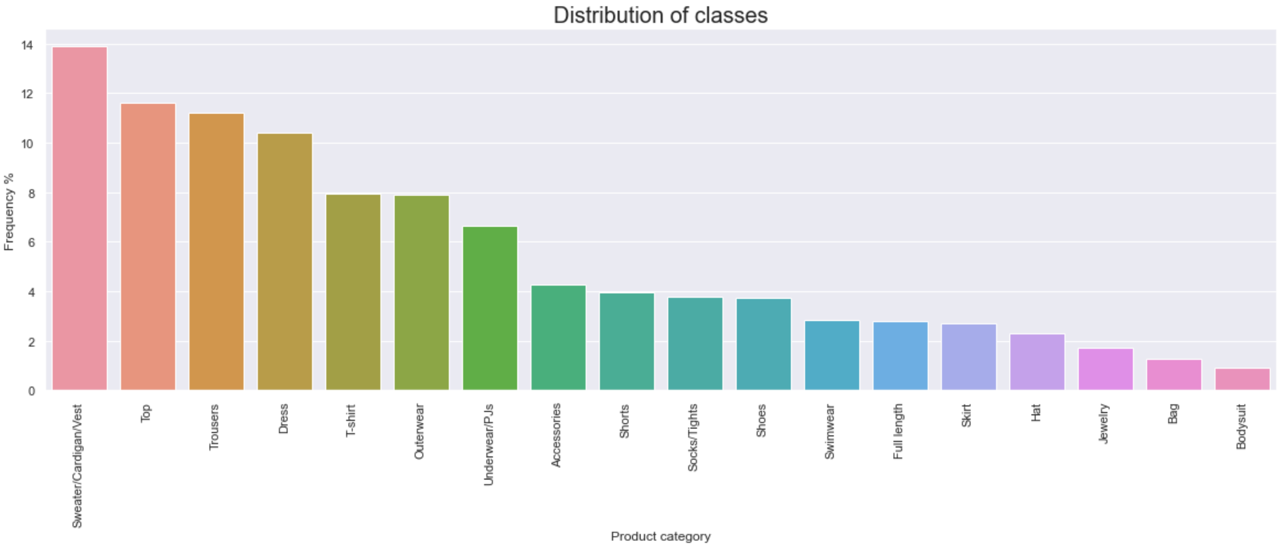


Figure 4.1 Bar graph showing the distribution of product categories

4.2 Potential source of data leakage

Investigate if there is a data leakage issue between product title/description and the target y variable: product category. Does this happen for every category? How often do they overlap?

4.2.1 Most frequently occuring words in the product title

There are 1,635,782 total words in the product title column `prod_name`. Plotting this on a word cloud shows many of these words overlap with the 20 product categories.

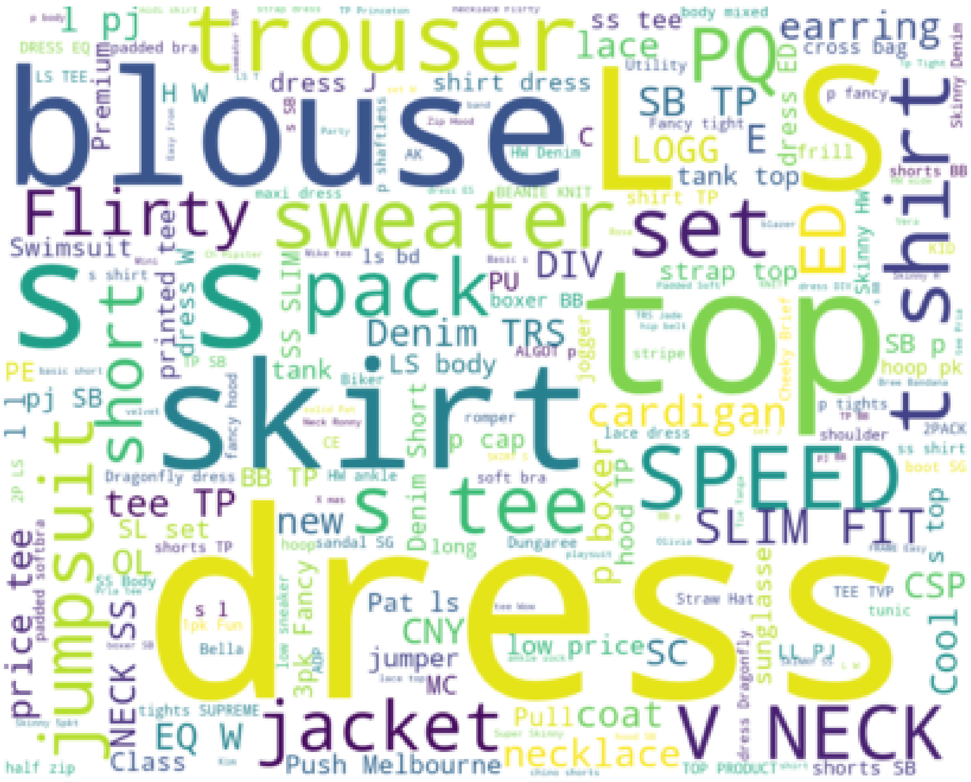


Figure 4.2.1 Word cloud of all the product titles

4.2.2 Most frequently occuring words in the product descriptions

There are 14,252,216 total words in the product description column `detail_desc`.

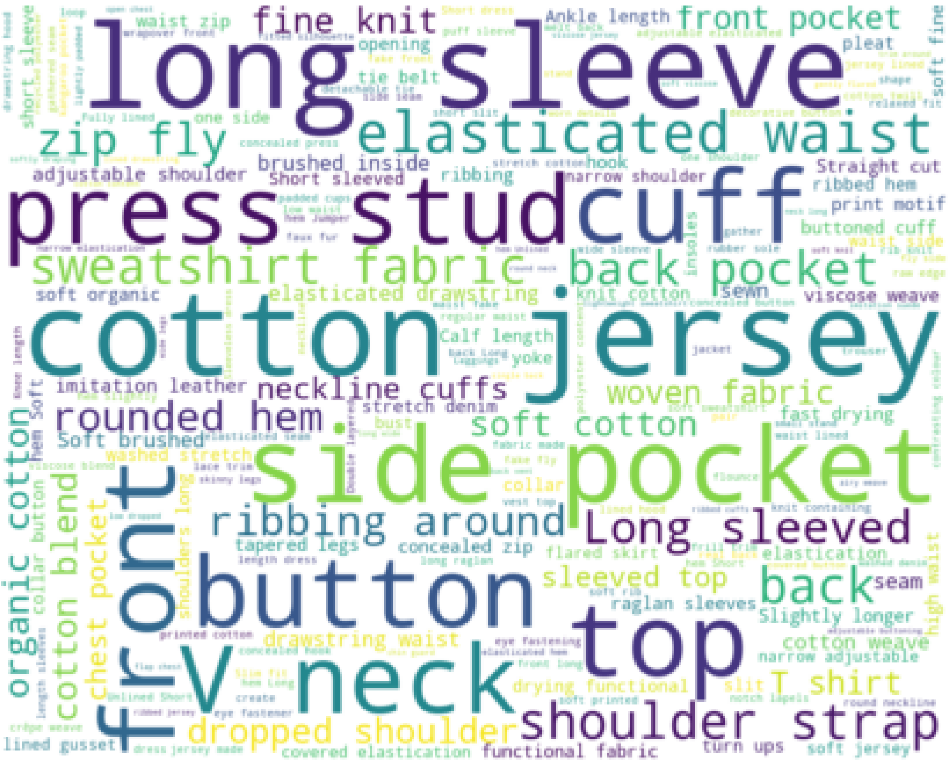


Figure 4.2.2 Word cloud of all the product descriptions

Word clouds of the most frequently occuring words in the product description shows more adjectives about fabric and clothing attributes than nouns to describe the product category. Product titles will have more information about the product category an item belongs to. The variety shown in product descriptions could apply to multiple different product categories and thus might not be as useful to the machine learning models.

Use stemming for category names and product titles to see the number of times they overlap. Plotting these as an absolute number and as a percentage on bar graphs is shown below.

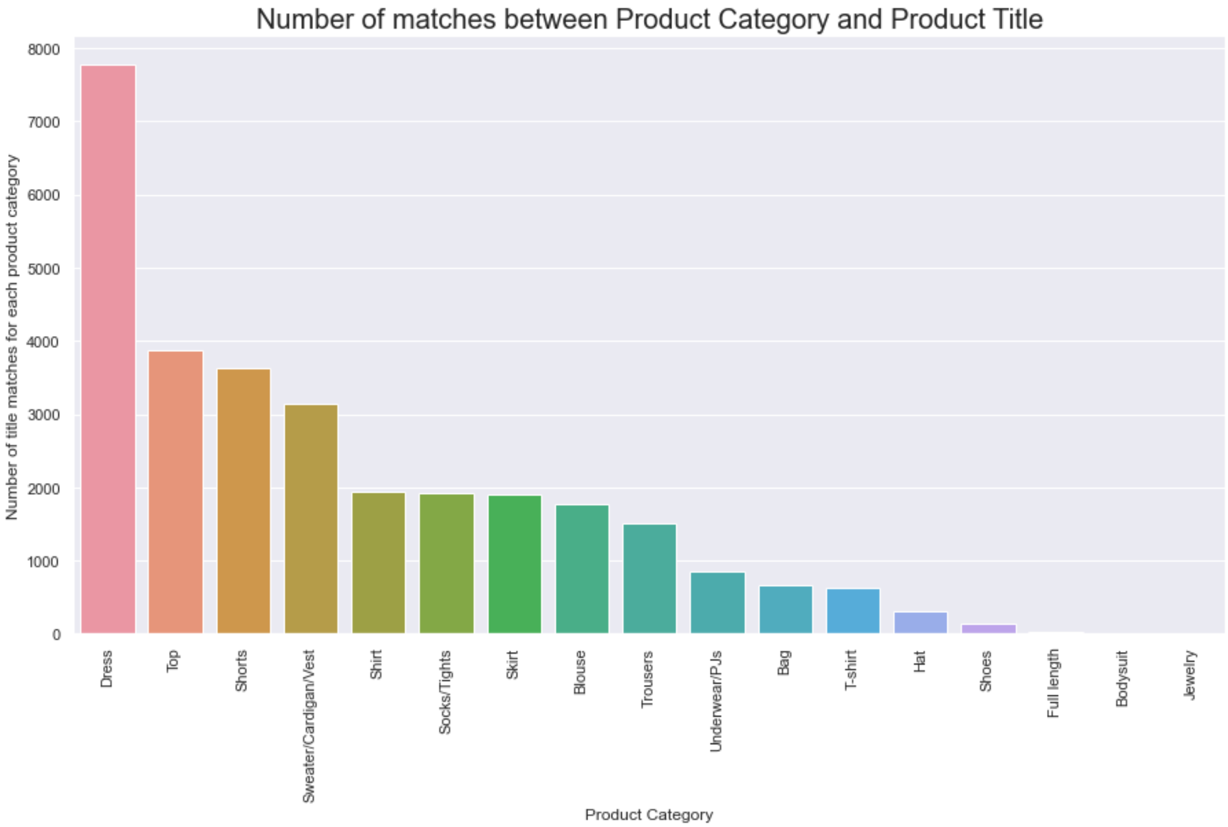


Figure 4.2.3 Bar graph showing the absolute number of matches between product titles and product categories

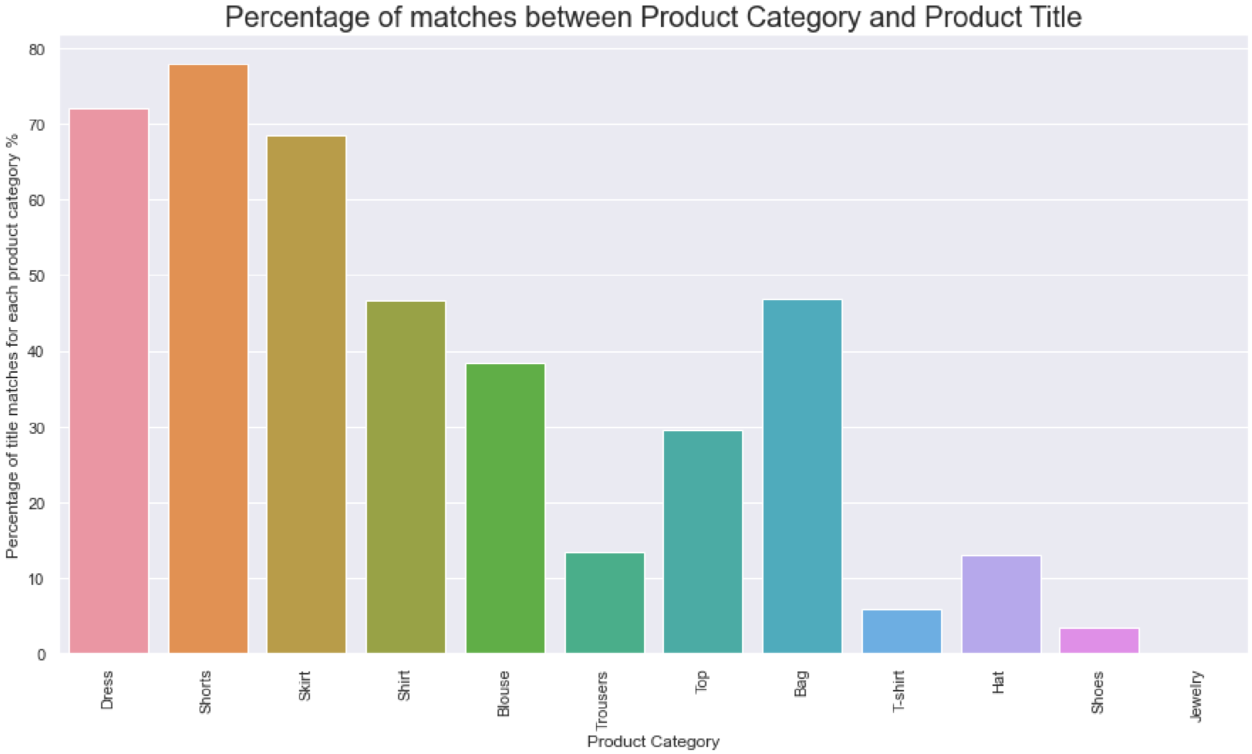


Figure 4.2.4 Bar graph showing the number of matches as a percentage between product titles and product categories

This bar graph shows that there are many product categories that have the name repeated in the product title. For some categories such as 'Shorts' this occurs more than 75% of the time, which is so high these categories could use it's own title as the algorithm for predicting product category. Other product categories such as Trousers are a lot less likely to have the word repeated in the product title.

I propose to build several models using image embeddings and/or product text (title and description) to determine which is the best combination of features to use. Categories with a high overlap between product title and category could use the results of this stemming word match as a baseline model.

4.3 Visualizing high dimensional image embedding data

Use t-SNE to do dimensionality reduction to 2, so the image embeddings can be visualized on a 2-D scatterplot.

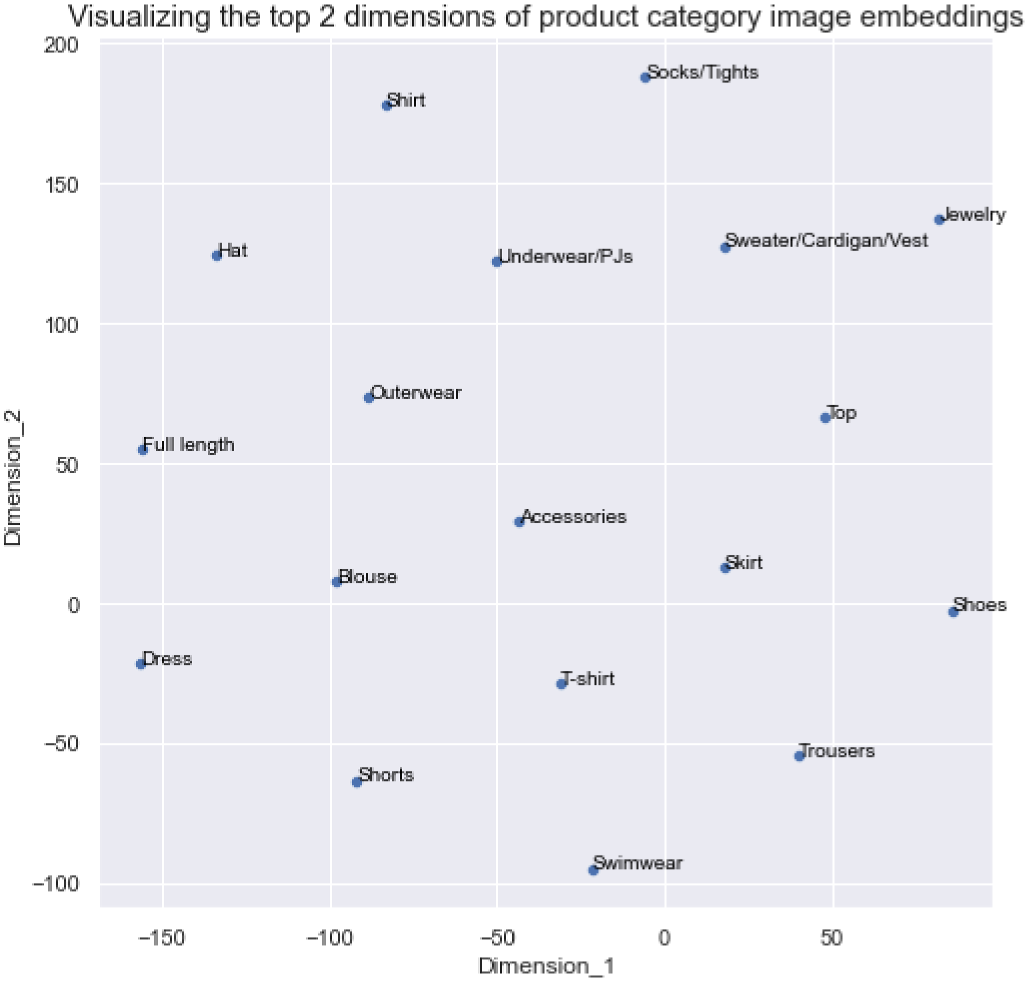


Figure 4.3.1 2D Scatter plot of the top 2 dimensions of product category image embeddings after dimensionality reduction

Use t-SNE to do dimensionality reduction to 3, so the image embeddings can be visualized on a 3-D scatterplot.

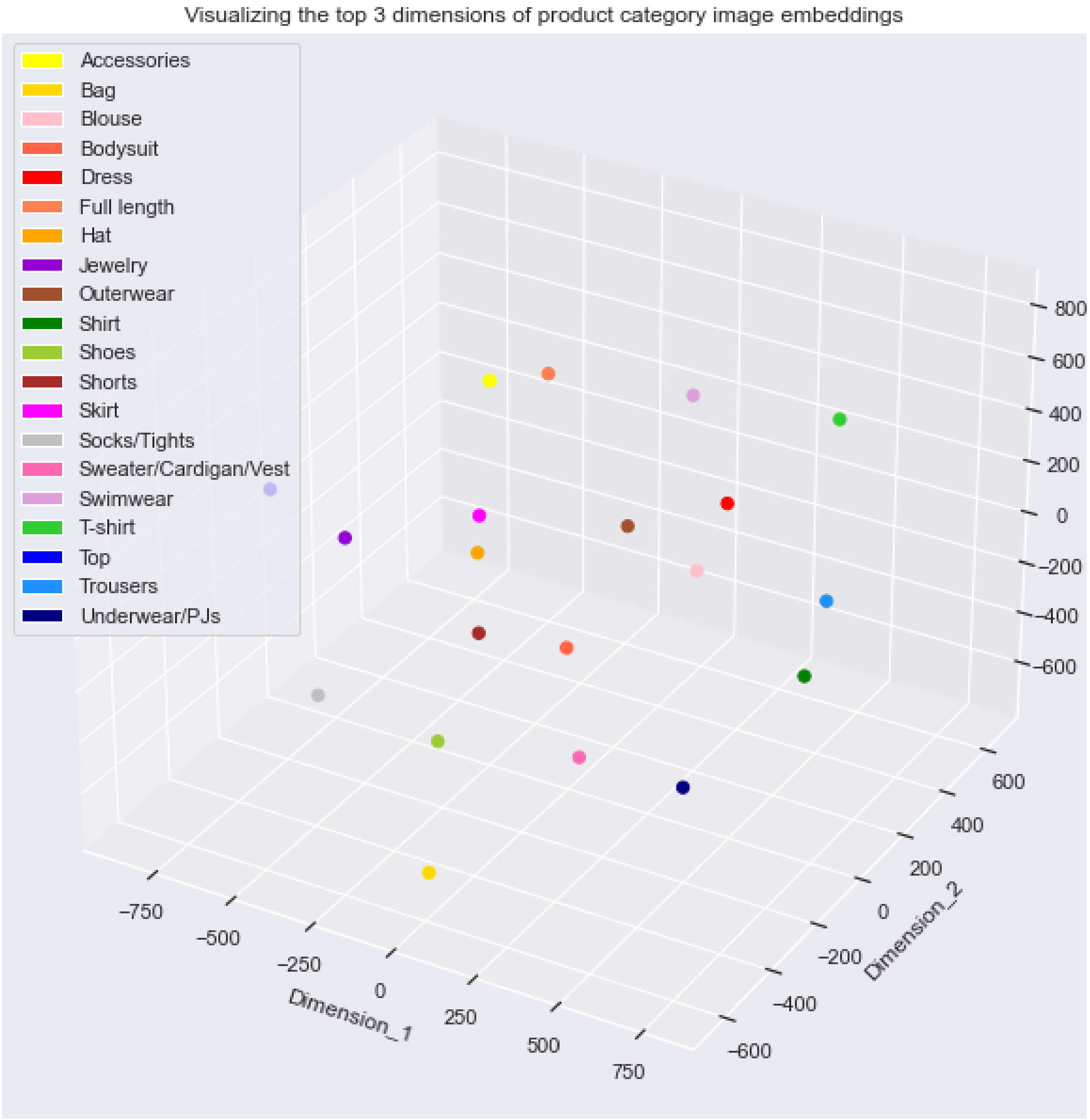


Figure 4.3.2 3D Scatter plot of the top 3 dimensions of product category image embeddings after dimensionality reduction

This 3D scatter plot shows some clustering for similar product categories. Accessories, Bag, Hat are on the left side, shoes and socks are close together, and Outerwear, sweater/cardigan/vest and full length are clustered in the middle.

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

Plot the distribution of Cosine Similarity for 1000 pairs of products

There are 3 groups to consider:
Group 1: Products that are the same but are color variants share the same 'product_code' and should have the most similar image embeddings.

Group 2: Products that are similar and in the same product category have different 'product_code' and the same 'product_type_name'.

Group 3: Products that are in different categories should have the least similarity between image embeddings.

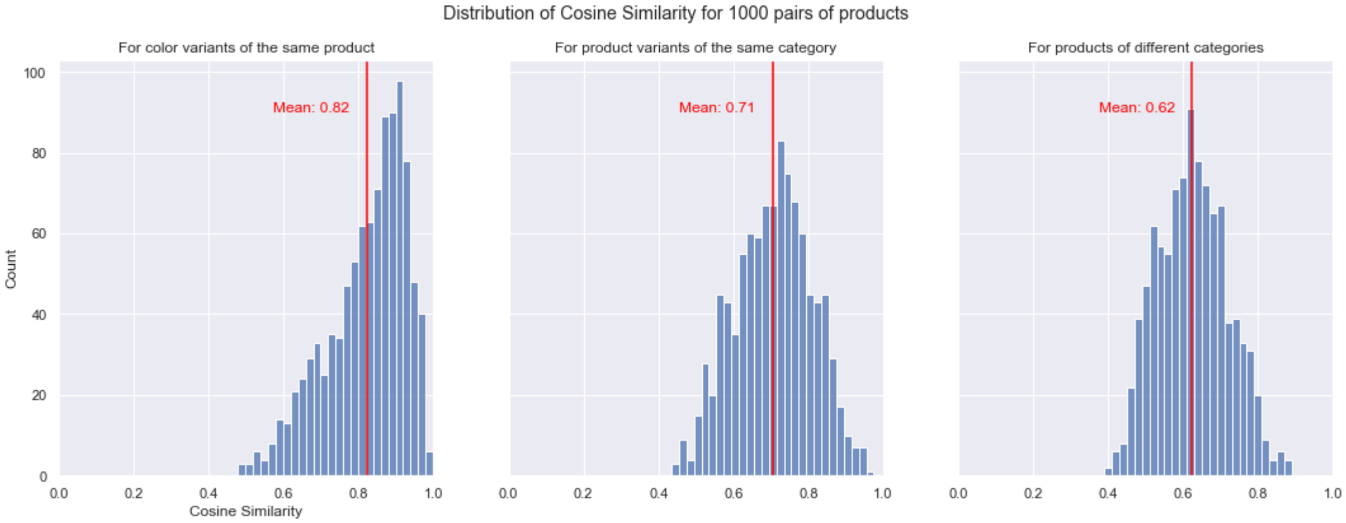


Figure 4.4 Bar graphs plotting the cosine similarity of 1000 pairs of products' image embeddings and their mean values by product category groups

As expected, Group 1 has the highest cosine similarity, and group 3 has the lowest.

5 Preprocessing and feature engineering

5.1 Preprocessing images - create image embeddings

Due to the large number of images, the image embeddings were created in a separate jupyter notebook and done in batches and combined into a single dataframe. See the notebook here: image embeddings notebook

ResNet-34 from the 'Deep Residual Learning for Image Recognition' research paper was used, along with the pretrained model weights. This convolutional neural net model includes up to 152 layers. Since the image embedding evaluation was done on a laptop, a simpler neural net model was chosen as other higher numbered ResNet models would have taken longer to evaluate.

The image preprocessing steps were setup to be the same as the training steps for the ResNet-34 model:

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

The image mode for this model is RGB, and if the image is Greyscale, it is converted to RGB.

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

5.2 Preprocessing NLP features

Using NLTK, create a bag of words from the product title and product description, excluding stop words, numbers and punctuation. A Counter is used to store the frequencies of each word occurrence, and words must appear a minimum of 5 times to be considered as a feature.

100 of the most frequently appearing words are chosen as features (where each word is a column, thus adding 100 NLP features to the dataset). Once the features are added, the NLP columns are populated with a 0 or 1 if the product title and description includes the feature word.

All of the numeric features are binary features so no scaling is required.

Other preprocessing steps:

1. Relabel misclassified product categories: spot checking the product categories indicates that some of the data is mislabeled, and should be corrected before modeling.
2. Remove the 397 image duplicates (where the image embeddings are exactly the same) but the product title/product description differs.

5.3 Group Shuffle Split

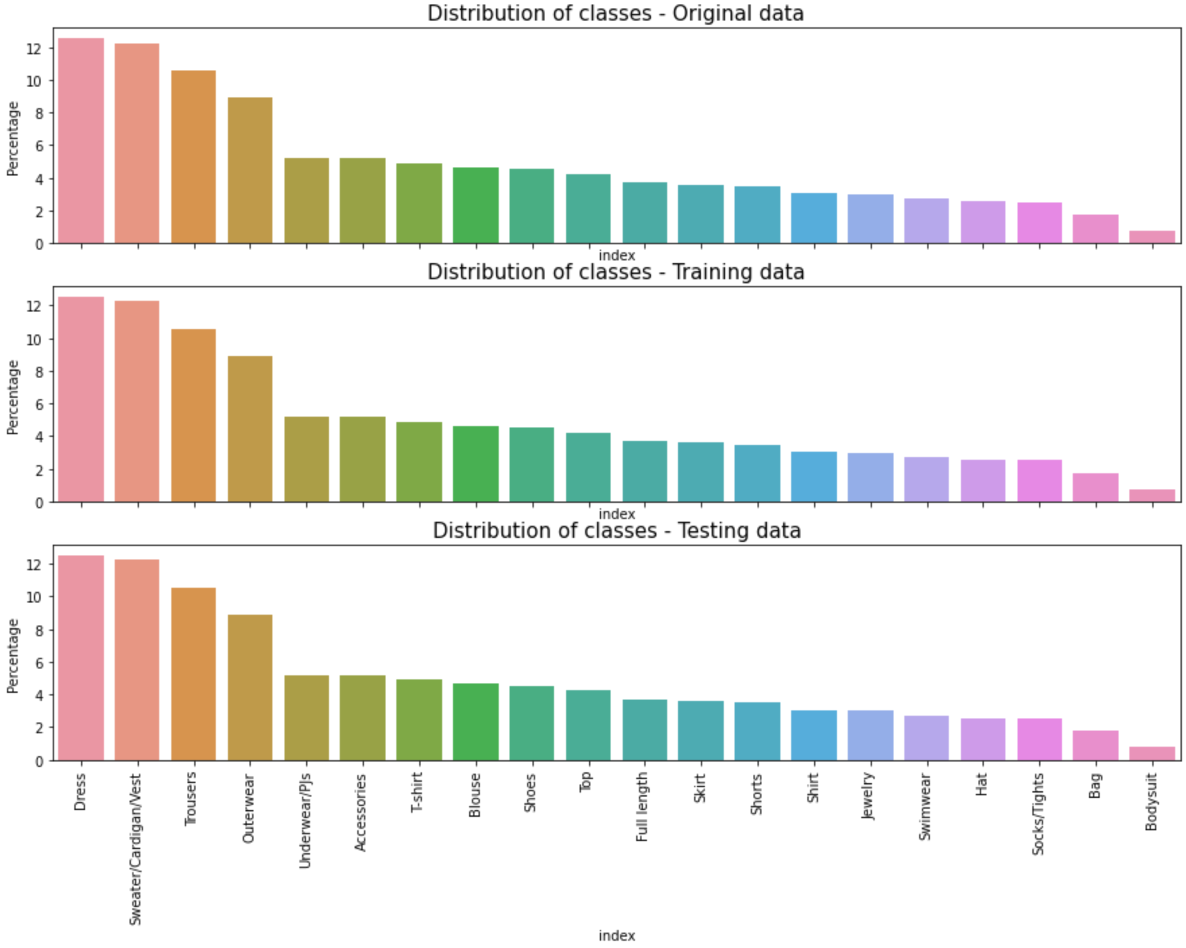
The model is trained on 70% of the data, with the remaining 30% used to evaluate performance of the models. The data frame has 99332 rows of products. Note in the data wrangling stage, product variants were kept in the dataset because they provided more rows of data. Typically the only difference between product variants (all product variants have the same column `product_code`) is color or pattern.

There are two key concerns with this dataset:

1. Training and testing split needs to be done grouped by product variants, otherwise data leakage will occur. I used SKLearn's GroupShuffleSplit which is a Shuffle-Group-Out cross-validation iterator.

2. This dataset has class imbalances, so the training and testing set needs to maintain the same distribution of class imbalances after splitting.

In order to stratify the training and testing groups to keep the class distributions, subset the data frame by each of the 20 major categories and perform group shuffle split before combining them together after the split. To check the imbalanced classes are maintained, bar graphs are plotted of the dataset before splitting, training and testing.



Distribution of classes - Original data

Distribution of classes - Training data

Distribution of classes - Testing data

Figure 5.3.1 Bar graphs showing the distribution of classes between the original data, training set and test set

6 Machine Learning Models

6.1 Summary of Machine Learning models

1. Baseline Model
2. Stochastic Gradient Descent Classifier using Elastic Net
3. Random Forest Classifier
4. KNN Classifier
5. Deep Neural Net using Keras

Machine Learning models 2, 3 and 4 used SKLearn's Randomized Search for hyperparameter tuning.

6.2 Baseline Model

A simple baseline model would be to use the imbalanced class distributions of each product category. Random guessing the most popular class category would see almost 14% accuracy.

Product Category	**% of Dataset**	**Product Category**	**% of Dataset**
Sweater/Cardigan/Vest	13.89	Shoes	3.81
Trousers	11.20	Socks/Tights	3.78
Dress	10.40	Shirt	3.42
T-shirt	7.93	Swimwear	2.83
Outerwear	7.89	Full length	2.78
Underwear/PJs	6.65	Skirt	2.70
Accessories	4.36	Hat	2.32
Top	4.12	Jewelry	1.75
Blouse	3.99	Bag	1.28
Shorts	3.95	Bodysuit	0.92

Table 6.2 Summary of product categories as a percentage of the entire dataset

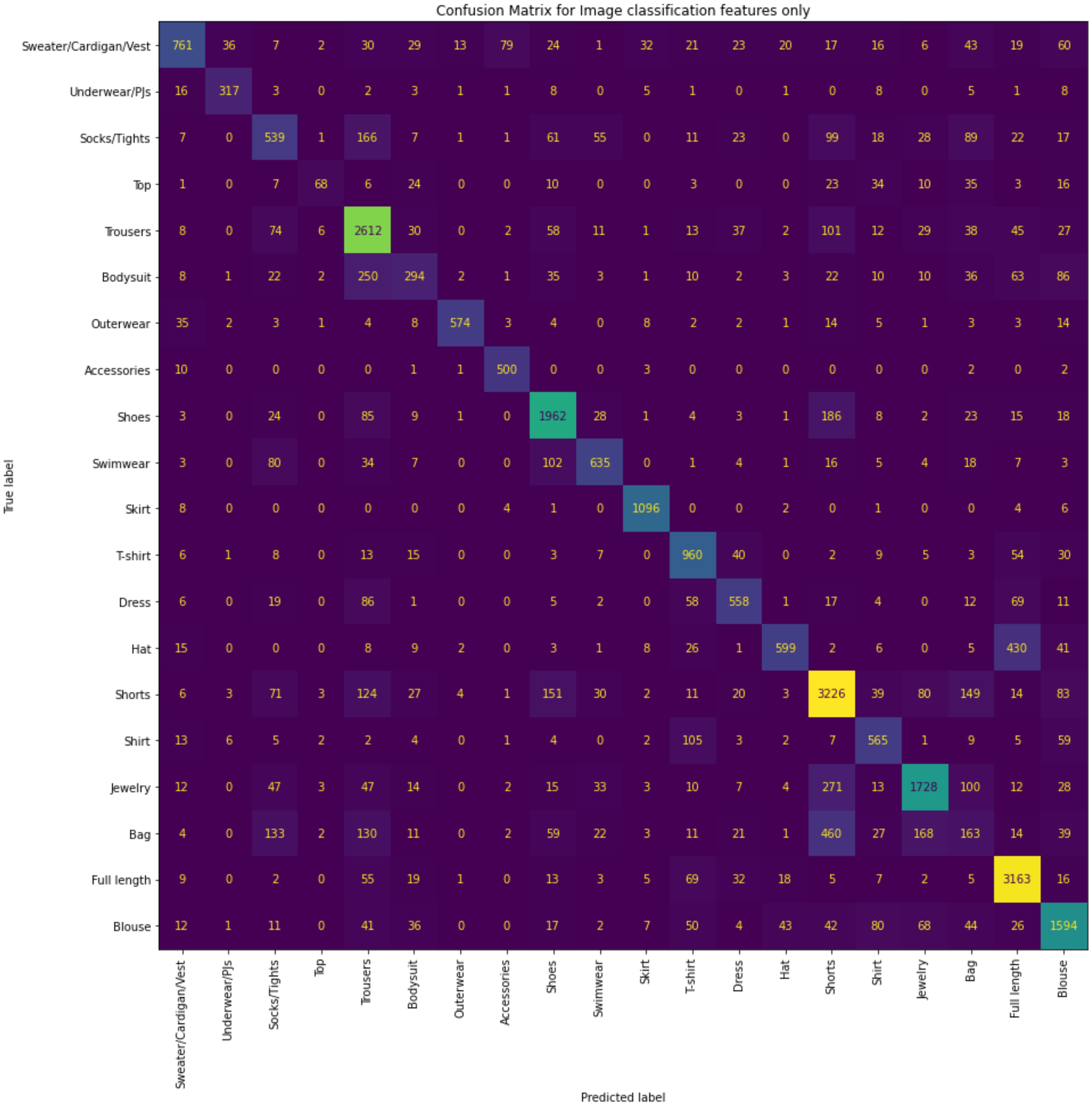
6.3 Elastic Net Linear model

Train a Stochastic gradient descent linear model using Randomized Search with 5 cross validation folds.
Plot the classification matrix and weighted f1-score as a way to compare performance of product categories between the machine learning models.

Different combinations of the following types of features were evaluated on the Linear model:

6.3.1 Image embedding features
6.3.2 Product title features
6.3.3 Product title + description features
6.3.4 Product title + image embeddings
6.3.5 Product title + description features + image embedding features

6.3.1 Image embedding features



	Sweater/Cardigan/Vest	Underwear/PJs	Socks/Tights	Top	Trousers	Bodysuit	Outerwear	Accessories	Shoes	Swimwear	Skirt	T-shirt	Dress	Hat	Shorts	Shirt	Jewelry	Bag	Full length	Blouse
Sweater/Cardigan/Vest	761	36	7	2	30	29	13	79	24	1	32	21	23	20	17	16	6	43	19	60
Underwear/PJs	16	317	3	0	2	3	1	1	8	0	5	1	0	1	0	8	0	5	1	8
Socks/Tights	7	0	539	1	166	7	1	1	61	55	0	11	23	0	99	18	28	89	22	17
Top	1	0	7	68	6	24	0	0	10	0	0	3	0	0	23	34	10	35	3	16
Trousers	8	0	74	6	2612	30	0	2	58	11	1	13	37	2	101	12	29	38	45	27
Bodysuit	8	1	22	2	250	294	2	1	35	3	1	10	2	3	22	10	10	36	63	86
Outerwear	35	2	3	1	4	8	574	3	4	0	8	2	2	1	14	5	1	3	3	14
Accessories	10	0	0	0	0	1	1	500	0	0	3	0	0	0	0	0	0	2	0	2
Shoes	3	0	24	0	85	9	1	0	1962	28	1	4	3	1	186	8	2	23	15	18
Swimwear	3	0	80	0	34	7	0	0	102	635	0	1	4	1	16	5	4	18	7	3
Skirt	8	0	0	0	0	0	0	4	1	0	1096	0	0	2	0	1	0	0	4	6
T-shirt	6	1	8	0	13	15	0	0	3	7	0	960	40	0	2	9	5	3	54	30
Dress	6	0	19	0	86	1	0	0	5	2	0	58	558	1	17	4	0	12	69	11
Hat	15	0	0	0	8	9	2	0	3	1	8	26	1	599	2	6	0	5	430	41
Shorts	6	3	71	3	124	27	4	1	151	30	2	11	20	3	3226	39	80	149	14	83
Shirt	13	6	5	2	2	4	0	1	4	0	2	105	3	2	7	565	1	9	5	59
Jewelry	12	0	47	3	47	14	0	2	15	33	3	10	7	4	271	13	1728	100	12	28
Bag	4	0	133	2	130	11	0	2	59	22	3	11	21	1	460	27	168	163	14	39
Full length	9	0	2	0	55	19	1	0	13	3	5	69	32	18	5	7	2	5	3163	16
Blouse	12	1	11	0	41	36	0	0	17	2	7	50	4	43	42	80	68	44	26	1594

Figure 6.3.1 Elastic Net Linear Model - Classification Matrix of Image embedding features showing predicted and true product category labels

Most of the classes are correctly predicted, but there are many smaller product categories related to accessories that are mislabeled. The top 3 classes are: Full length, Blouse, Skirt. The worst performing classes are: Jewelry, Bag, Hat. An inspection of the misclassified products shows that the image embeddings model does not perform well when there are close up images of the product instead of a view of the entire product. Another issue this model struggles with is kids clothing even when the image is of the entire product - especially kids trousers that are labeled as shorts. I suspect the proportion of the leg length is similar to adult shorts. Adding product titles/description features should help counter this issue.

6.3.2 Product title features

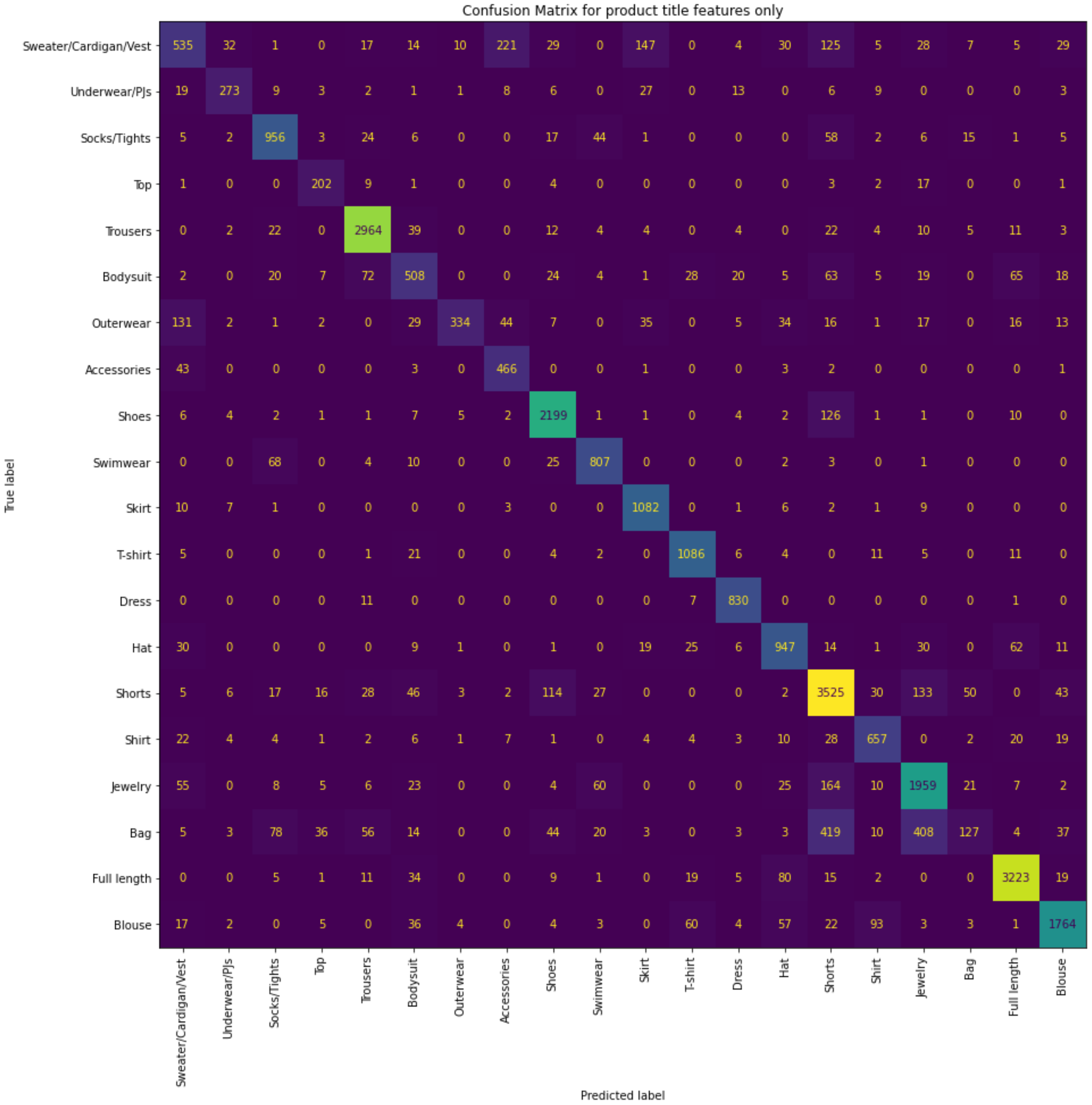
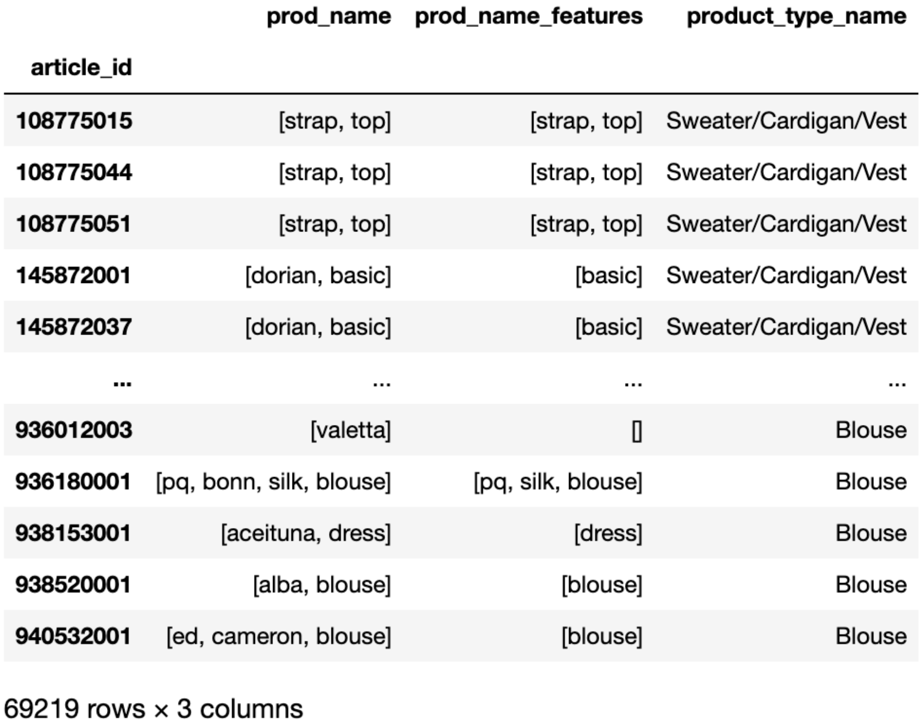


Figure 6.3.2.1 Elastic Net Linear Model - Classification Matrix of product title NLP features showing predicted and true product category labels

Top classes: Trousers, Shorts, Full Length, Dress
Worst performing 3 classes: Sweater/Cardigan/Vest, Jewelry, Bag
A comparison of these results to the bar plot showing the percentage of overlap between product category and product title (figure 4.2.4) suggests the model is not performing as well as it should. Since more than 45% of product titles for the product category Bags contain the word ‘bag’, I would expect the model to predict a higher accuracy.

An inspection of the NLP features generated from X_train's product titles shows:



	prod_name	prod_name_features	product_type_name
article_id			
108775015	[strap, top]	[strap, top]	Sweater/Cardigan/Vest
108775044	[strap, top]	[strap, top]	Sweater/Cardigan/Vest
108775051	[strap, top]	[strap, top]	Sweater/Cardigan/Vest
145872001	[dorian, basic]	[basic]	Sweater/Cardigan/Vest
145872037	[dorian, basic]	[basic]	Sweater/Cardigan/Vest
...	...	...	...
936012003	[valetta]	[]	Blouse
936180001	[pq, bonn, silk, blouse]	[pq, silk, blouse]	Blouse
938153001	[aceituna, dress]	[dress]	Blouse
938520001	[alba, blouse]	[blouse]	Blouse
940532001	[ed, cameron, blouse]	[blouse]	Blouse

69219 rows × 3 columns

Figure 6.3.2.2 Snapshot of NLP features created from X_train product titles

Where 'prod_name' is the product's title, 'prod_name_features' are the NLP features created in the preprocessing pipeline, and 'product_type_name' is the product category the model is trying to predict.

Since the title features are created from the top 100 words in a sorted dictionary of product titles, whether or not a feature is created depends on its popularity. If a product title name is less frequently occurring then it will have worse features or even no features. In this example, the product title 'valetta' does not produce any features.

The percentage of product titles that have 1 or more features created from this NLP pipeline is 64%, meaning 36% of products do not have features. This shows the need for images/supplemental data.

Looking at the feature ‘basic’ vs. the class distribution of this feature shows it appears across many product categories and more data is needed for this model to improve.

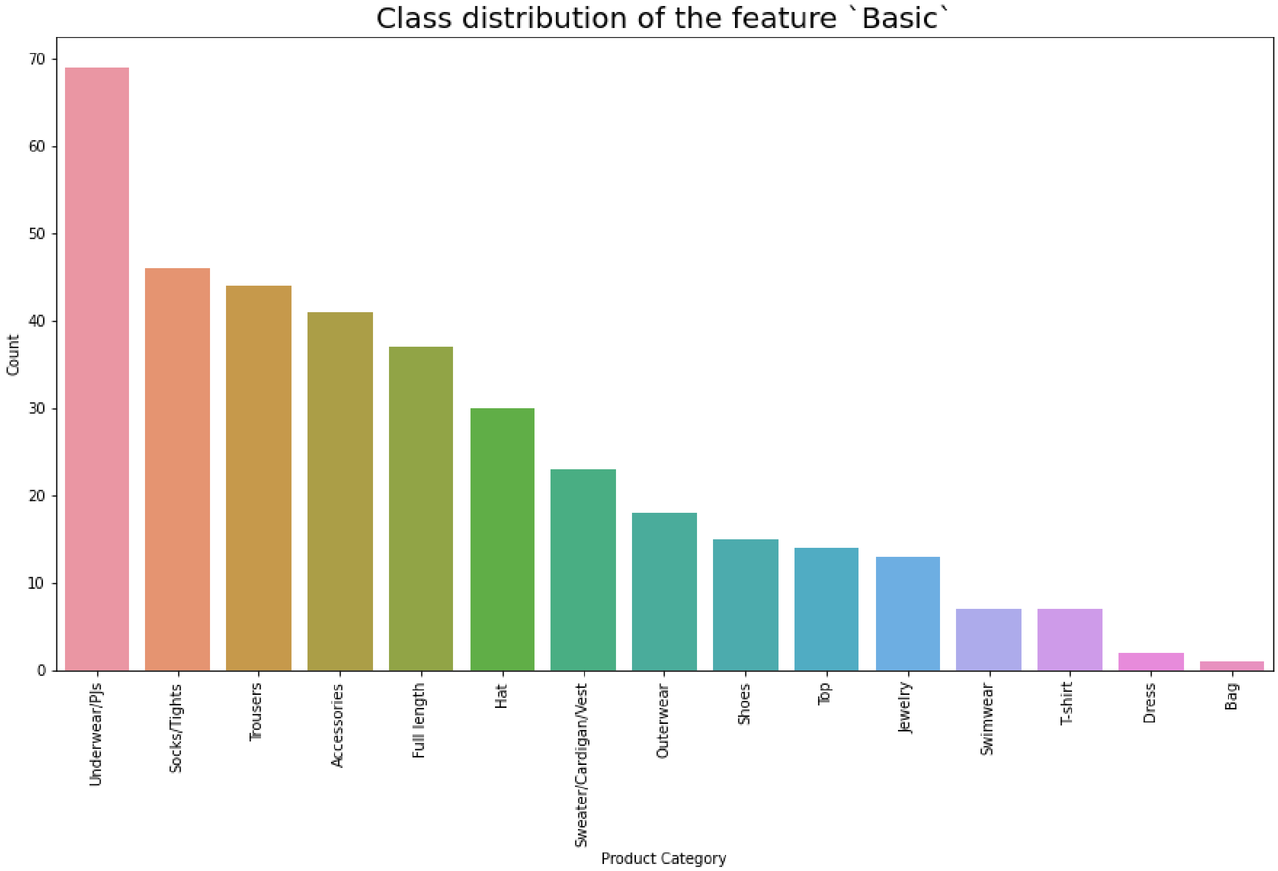
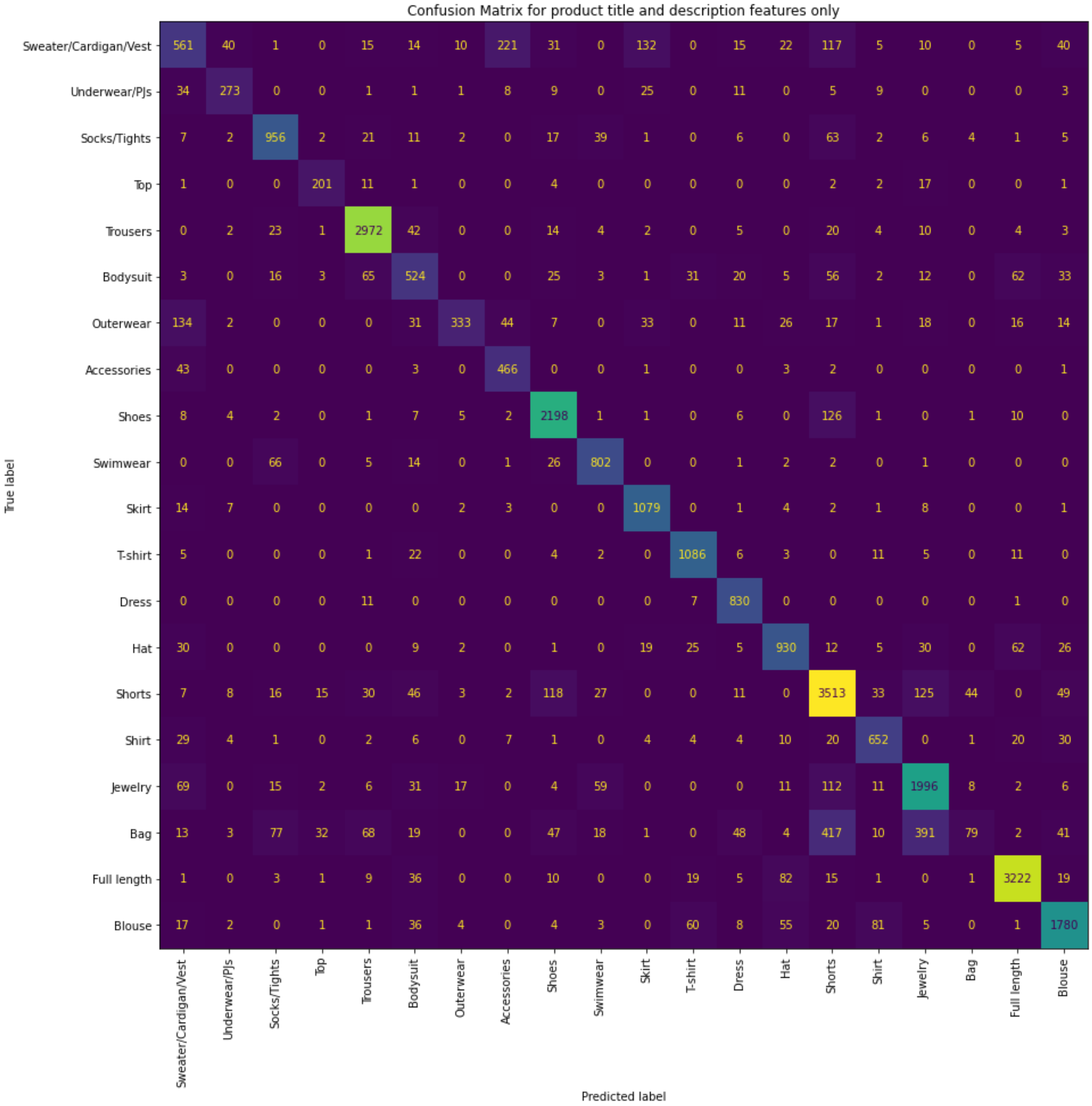


Figure 6.3.2.3 Bar graph showing the class distribution of the NLP feature ‘Basic’

6.3.3 Product title + description features



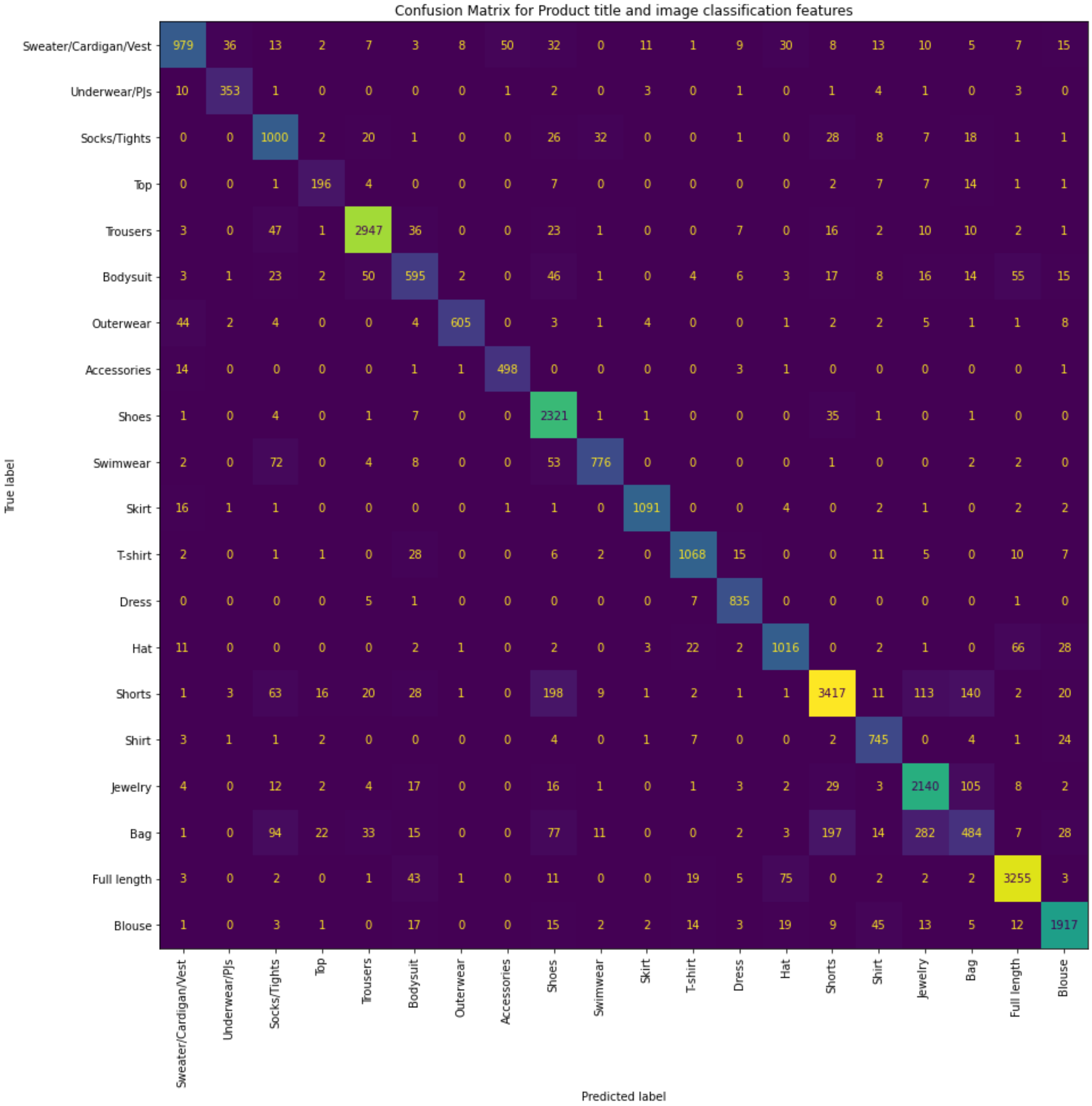
	Sweater/Cardigan/Vest	Underwear/PJs	Socks/Tights	Top	Trousers	Bodysuit	Outerwear	Accessories	Shoes	Swimwear	Skirt	T-shirt	Dress	Hat	Shorts	Shirt	Jewelry	Bag	Full length	Blouse
Sweater/Cardigan/Vest	561	40	1	0	15	14	10	221	31	0	132	0	15	22	117	5	10	0	5	40
Underwear/PJs	34	273	0	0	1	1	1	8	9	0	25	0	11	0	5	9	0	0	0	3
Socks/Tights	7	2	956	2	21	11	2	0	17	39	1	0	6	0	63	2	6	4	1	5
Top	1	0	0	201	11	1	0	0	4	0	0	0	0	0	2	2	17	0	0	1
Trousers	0	2	23	1	2972	42	0	0	14	4	2	0	5	0	20	4	10	0	4	3
Bodysuit	3	0	16	3	65	524	0	0	25	3	1	31	20	5	56	2	12	0	62	33
Outerwear	134	2	0	0	0	31	333	44	7	0	33	0	11	26	17	1	18	0	16	14
Accessories	43	0	0	0	0	3	0	466	0	0	1	0	0	3	2	0	0	0	0	1
Shoes	8	4	2	0	1	7	5	2	2198	1	1	0	6	0	126	1	0	1	10	0
Swimwear	0	0	66	0	5	14	0	1	26	802	0	0	1	2	2	0	1	0	0	0
Skirt	14	7	0	0	0	0	2	3	0	0	1079	0	1	4	2	1	8	0	0	1
T-shirt	5	0	0	0	1	22	0	0	4	2	0	1086	6	3	0	11	5	0	11	0
Dress	0	0	0	0	11	0	0	0	0	0	0	7	830	0	0	0	0	0	1	0
Hat	30	0	0	0	0	9	2	0	1	0	19	25	5	930	12	5	30	0	62	26
Shorts	7	8	16	15	30	46	3	2	118	27	0	0	11	0	3513	33	125	44	0	49
Shirt	29	4	1	0	2	6	0	7	1	0	4	4	4	10	20	652	0	1	20	30
Jewelry	69	0	15	2	6	31	17	0	4	59	0	0	0	11	112	11	1996	8	2	6
Bag	13	3	77	32	68	19	0	0	47	18	1	0	48	4	417	10	391	79	2	41
Full length	1	0	3	1	9	36	0	0	10	0	0	19	5	82	15	1	0	1	3222	19
Blouse	17	2	0	1	1	36	4	0	4	3	0	60	8	55	20	81	5	0	1	1780

Figure 6.3.3 Elastic Net Linear Model - Classification Matrix of product title and description NLP features showing predicted and true product category labels

Top classes: Trousers, Full Length, Dress, Blouse
Worst performing 3 classes: Sweater/Cardigan/Vest, Bag, Shorts

This model is an improvement from the product title features model. The class Jewelry has improved, probably because the product description adds more useful features. Note that Jewelry as a product title only had 1 match to its category name. I suspect this is because most Jewelry would be labeled in more detail e.g. earrings and not labeled as genetically as Jewelry.

6.3.4 Product title + image embeddings



Confusion Matrix for Product title and image classification features

True label

	Sweater/Cardigan/Vest	Underwear/PJs	Socks/Tights	Top	Trousers	Bodysuit	Outerwear	Accessories	Shoes	Swimwear	Skirt	T-shirt	Dress	Hat	Shorts	Shirt	Jewelry	Bag	Full length	Blouse
Sweater/Cardigan/Vest	979	36	13	2	7	3	8	50	32	0	11	1	9	30	8	13	10	5	7	15
Underwear/PJs	10	353	1	0	0	0	0	1	2	0	3	0	1	0	1	4	1	0	3	0
Socks/Tights	0	0	1000	2	20	1	0	0	26	32	0	0	1	0	28	8	7	18	1	1
Top	0	0	1	196	4	0	0	0	7	0	0	0	0	0	2	7	7	14	1	1
Trousers	3	0	47	1	2947	36	0	0	23	1	0	0	7	0	16	2	10	10	2	1
Bodysuit	3	1	23	2	50	595	2	0	46	1	0	4	6	3	17	8	16	14	55	15
Outerwear	44	2	4	0	0	4	605	0	3	1	4	0	0	1	2	2	5	1	1	8
Accessories	14	0	0	0	0	1	1	498	0	0	0	0	3	1	0	0	0	0	0	1
Shoes	1	0	4	0	1	7	0	0	2321	1	1	0	0	0	35	1	0	1	0	0
Swimwear	2	0	72	0	4	8	0	0	53	776	0	0	0	0	1	0	0	2	2	0
Skirt	16	1	1	0	0	0	0	1	1	0	1091	0	0	4	0	2	1	0	2	2
T-shirt	2	0	1	1	0	28	0	0	6	2	0	1068	15	0	0	11	5	0	10	7
Dress	0	0	0	0	5	1	0	0	0	0	0	7	835	0	0	0	0	0	1	0
Hat	11	0	0	0	0	2	1	0	2	0	3	22	2	1016	0	2	1	0	66	28
Shorts	1	3	63	16	20	28	1	0	198	9	1	2	1	1	3417	11	113	140	2	20
Shirt	3	1	1	2	0	0	0	0	4	0	1	7	0	0	2	745	0	4	1	24
Jewelry	4	0	12	2	4	17	0	0	16	1	0	1	3	2	29	3	2140	105	8	2
Bag	1	0	94	22	33	15	0	0	77	11	0	0	2	3	197	14	282	484	7	28
Full length	3	0	2	0	1	43	1	0	11	0	0	19	5	75	0	2	2	2	3255	3
Blouse	1	0	3	1	0	17	0	0	15	2	2	14	3	19	9	45	13	5	12	1917

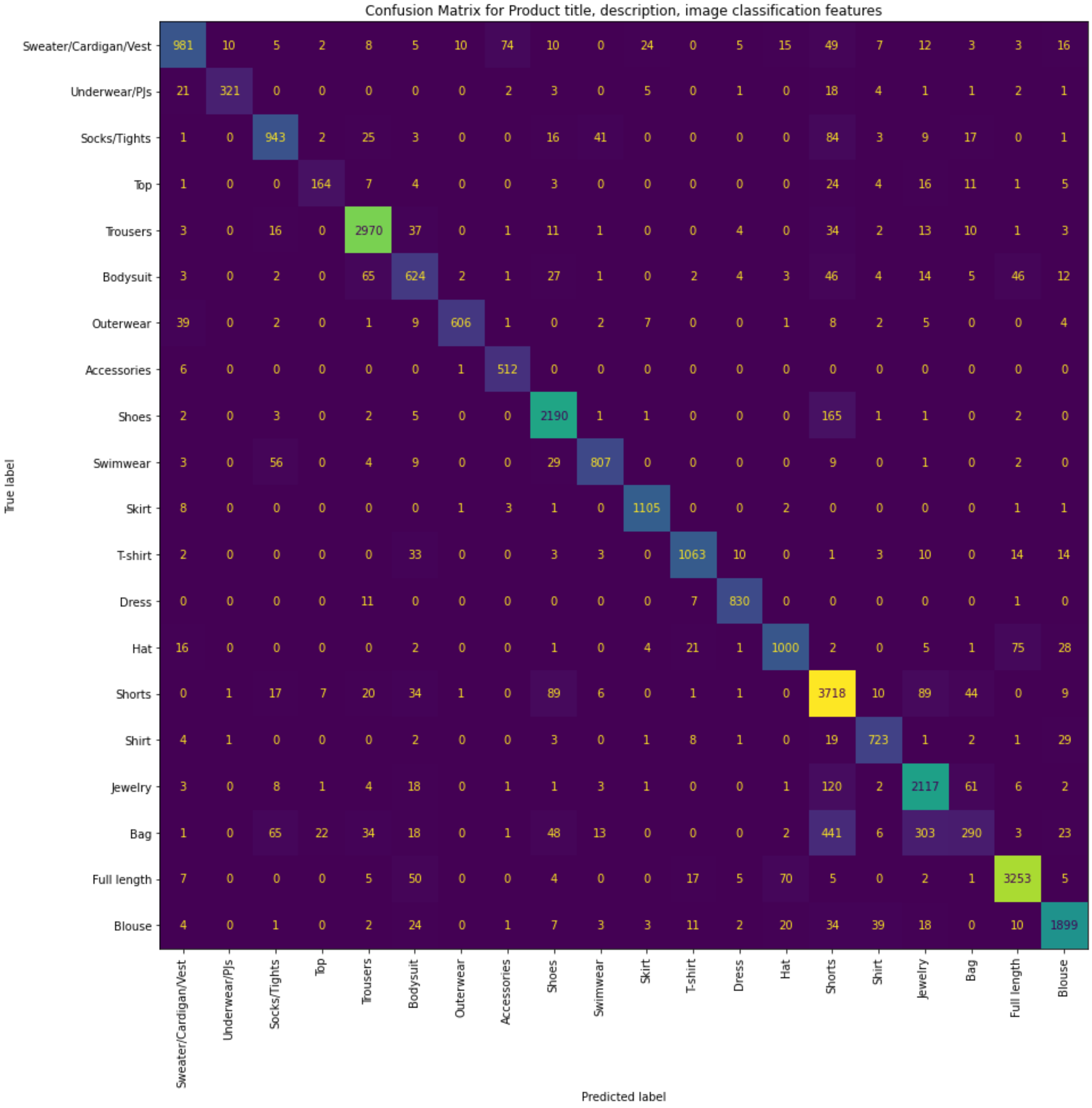
Predicted label

Figure 6.3.4 Elastic Net Linear Model - Classification Matrix of product title and image embedding features showing predicted and true product category labels

Top classes: Trousers, Shoes, Full Length, Dress, Blouse, Skirt
Worst performing 3 classes: Shorts, Bag, Jewelry

Most classes are performing very well. The classes that the model is having trouble predicting are still the same combination: Shorts, Bag, Jewelry.

6.3.5 Product title + description features + image embedding features



	Sweater/Cardigan/Vest	Underwear/PJs	Socks/Tights	Top	Trousers	Bodysuit	Outerwear	Accessories	Shoes	Swimwear	Skirt	T-shirt	Dress	Hat	Shorts	Shirt	Jewelry	Bag	Full length	Blouse
Sweater/Cardigan/Vest	981	10	5	2	8	5	10	74	10	0	24	0	5	15	49	7	12	3	3	16
Underwear/PJs	21	321	0	0	0	0	0	2	3	0	5	0	1	0	18	4	1	1	2	1
Socks/Tights	1	0	943	2	25	3	0	0	16	41	0	0	0	0	84	3	9	17	0	1
Top	1	0	0	164	7	4	0	0	3	0	0	0	0	0	24	4	16	11	1	5
Trousers	3	0	16	0	2970	37	0	1	11	1	0	0	4	0	34	2	13	10	1	3
Bodysuit	3	0	2	0	65	624	2	1	27	1	0	2	4	3	46	4	14	5	46	12
Outerwear	39	0	2	0	1	9	606	1	0	2	7	0	0	1	8	2	5	0	0	4
Accessories	6	0	0	0	0	0	1	512	0	0	0	0	0	0	0	0	0	0	0	0
Shoes	2	0	3	0	2	5	0	0	2190	1	1	0	0	0	165	1	1	0	2	0
Swimwear	3	0	56	0	4	9	0	0	29	807	0	0	0	0	9	0	1	0	2	0
Skirt	8	0	0	0	0	0	1	3	1	0	1105	0	0	2	0	0	0	0	1	1
T-shirt	2	0	0	0	0	33	0	0	3	3	0	1063	10	0	1	3	10	0	14	14
Dress	0	0	0	0	11	0	0	0	0	0	0	7	830	0	0	0	0	0	1	0
Hat	16	0	0	0	0	2	0	0	1	0	4	21	1	1000	2	0	5	1	75	28
Shorts	0	1	17	7	20	34	1	0	89	6	0	1	1	0	3718	10	89	44	0	9
Shirt	4	1	0	0	0	2	0	0	3	0	1	8	1	0	19	723	1	2	1	29
Jewelry	3	0	8	1	4	18	0	1	1	3	1	0	0	1	120	2	2117	61	6	2
Bag	1	0	65	22	34	18	0	1	48	13	0	0	0	2	441	6	303	290	3	23
Full length	7	0	0	0	5	50	0	0	4	0	0	17	5	70	5	0	2	1	3253	5
Blouse	4	0	1	0	2	24	0	1	7	3	3	11	2	20	34	39	18	0	10	1899

Figure 6.3.5 Elastic Net Linear Model - Classification Matrix of product title, description and image embedding features showing predicted and true product category labels

Top classes: The same as in the product title + image embeddings model
Worst performing 3 classes: Shoes, Shorts, Bag, Jewelry

This model performs the same as the product title + image model, despite having more data provided by the product descriptions. It's interesting to see the Jewelry class hasn't improved.

6.4 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. The weighted-averaged F1 score is calculated by taking the mean of all per-class F1 scores while considering each class's support. Since this dataset has class imbalances we want to assign greater contribution to classes with more examples in the dataset, thus the weighted average is used.

Table 6.4 Summary of weighted average f1 scores for different Linear model feature combinations

Feature combinations	**Weighted average F1-score**
Image	0.727
Product title	0.808
Product title + product description	0.806
Product title + Image classification	0.880
Product title + product description + image	0.872

These combinations show that classification using image embeddings only performs the worst. Product title performs about the same as product title + description and that product description doesn't add much additional information to the model.

Product title + description + images performs the same as title + images, so to simplify the model and reduce dimensionality, drop the description features.

Subsequent models will use product title and image embeddings only.

6.5 Random Forest Classifier

Training a Random Forest Classifier model using Randomized Search with 5 cross validation fold. The weighted average f1-score was 0.85

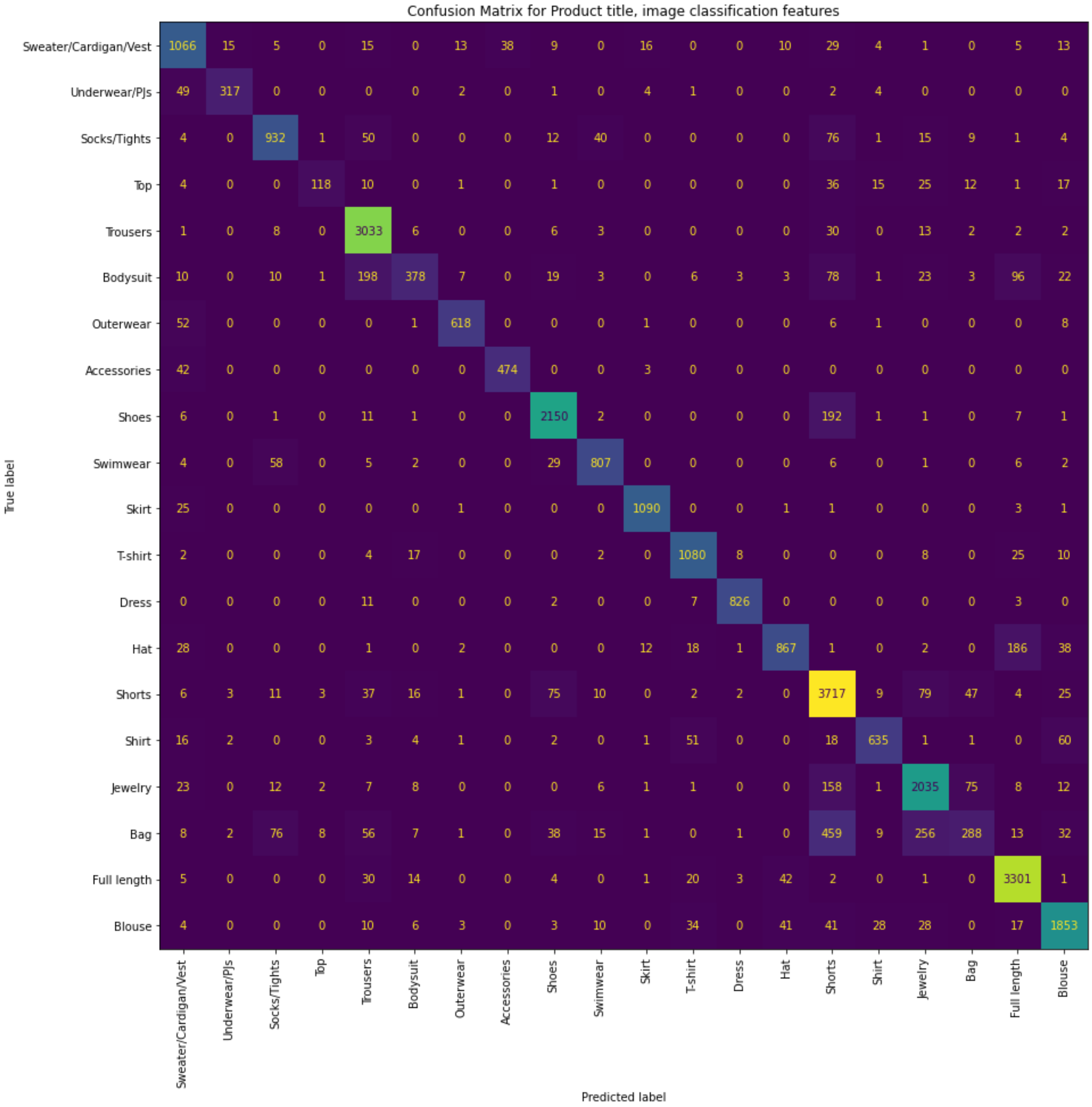


Figure 6.5.1 Random Forest Classification Matrix showing the predicted label and the true product category label

Top classes: Trousers, shorts, Full length
Worst performing classes: Bag, Jewelry, Shoes

Comparing the Random Forest model to the Linear model, the Random Forest model is producing some strange predictions:

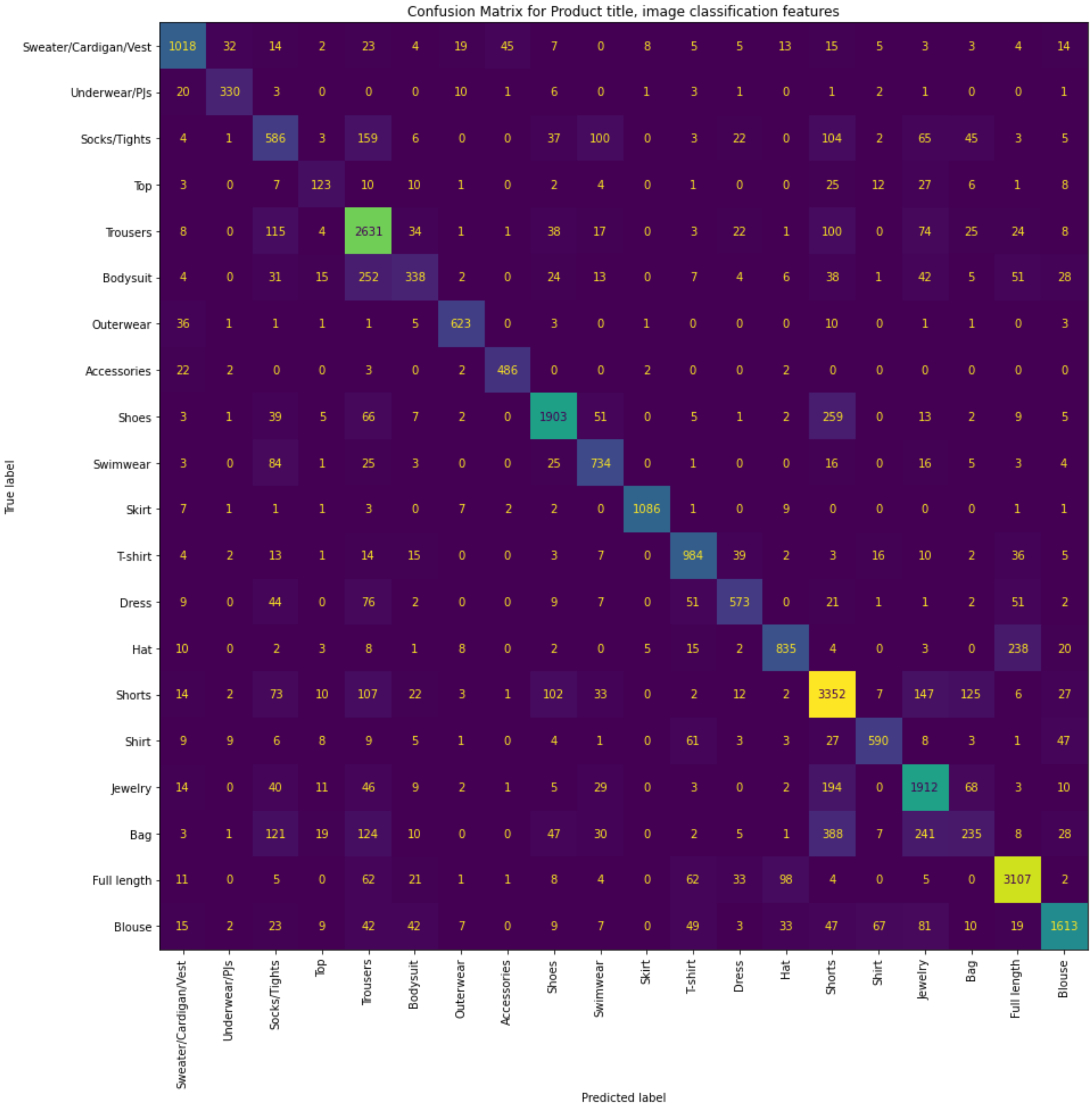
- Hat is being misclassified for Full length
- Bag is misclassified for Shorts, Jewelry
- Shoes is misclassified for Shorts

Inspecting the incorrect model predictions where the model was most confident shows the following insights:

- The majority of incorrect predictions are hoodies being labeled as Outerwear vs. the correct label of Sweater/Cardigan/Vest. This is a gray area and indicates the two product categories could be merged since they are so similar.
- Some other examples of tops are predicted to be tops in the model but the original dataset is Sweater/Cardigan/Vest which is a data labeling error that should be fixed.

6.6 KNN Classifier

Training a KNN model using Randomized Search with 5 cross validation folds.
The weighted average f1-score was 0.77



	Sweater/Cardigan/Vest	Underwear/PJs	Socks/Tights	Top	Trousers	Bodysuit	Outerwear	Accessories	Shoes	Swimwear	Skirt	T-shirt	Dress	Hat	Shorts	Shirt	Jewelry	Bag	Full length	Blouse
Sweater/Cardigan/Vest	1018	32	14	2	23	4	19	45	7	0	8	5	5	13	15	5	3	3	4	14
Underwear/PJs	20	330	3	0	0	0	10	1	6	0	1	3	1	0	1	2	1	0	0	1
Socks/Tights	4	1	586	3	159	6	0	0	37	100	0	3	22	0	104	2	65	45	3	5
Top	3	0	7	123	10	10	1	0	2	4	0	1	0	0	25	12	27	6	1	8
Trousers	8	0	115	4	2631	34	1	1	38	17	0	3	22	1	100	0	74	25	24	8
Bodysuit	4	0	31	15	252	338	2	0	24	13	0	7	4	6	38	1	42	5	51	28
Outerwear	36	1	1	1	1	5	623	0	3	0	1	0	0	0	10	0	1	1	0	3
Accessories	22	2	0	0	3	0	2	486	0	0	2	0	0	2	0	0	0	0	0	0
Shoes	3	1	39	5	66	7	2	0	1903	51	0	5	1	2	259	0	13	2	9	5
Swimwear	3	0	84	1	25	3	0	0	25	734	0	1	0	0	16	0	16	5	3	4
Skirt	7	1	1	1	3	0	7	2	2	0	1086	1	0	9	0	0	0	0	1	1
T-shirt	4	2	13	1	14	15	0	0	3	7	0	984	39	2	3	16	10	2	36	5
Dress	9	0	44	0	76	2	0	0	9	7	0	51	573	0	21	1	1	2	51	2
Hat	10	0	2	3	8	1	8	0	2	0	5	15	2	835	4	0	3	0	238	20
Shorts	14	2	73	10	107	22	3	1	102	33	0	2	12	2	3352	7	147	125	6	27
Shirt	9	9	6	8	9	5	1	0	4	1	0	61	3	3	27	590	8	3	1	47
Jewelry	14	0	40	11	46	9	2	1	5	29	0	3	0	2	194	0	1912	68	3	10
Bag	3	1	121	19	124	10	0	0	47	30	0	2	5	1	388	7	241	235	8	28
Full length	11	0	5	0	62	21	1	1	8	4	0	62	33	98	4	0	5	0	3107	2
Blouse	15	2	23	9	42	42	7	0	9	7	0	49	3	33	47	67	81	10	19	1613

Figure 6.6.1 KNN Classification Matrix showing the predicted label and the true product category label

The top 3 classes that performed well are: Skirt, Dress, Outerwear
The top 3 classes that performed the worst are: Shorts, Socks/Tights, Bag

Overall this KNN model performs worse than the SDGC or the Random Forest model. It has the same issues with the problem classes too.
Underwear/PJs performed better, and the model doesn't seem to confuse the product for T-shirts or Trousers, which the SDGC model with image embedding features only had trouble with.

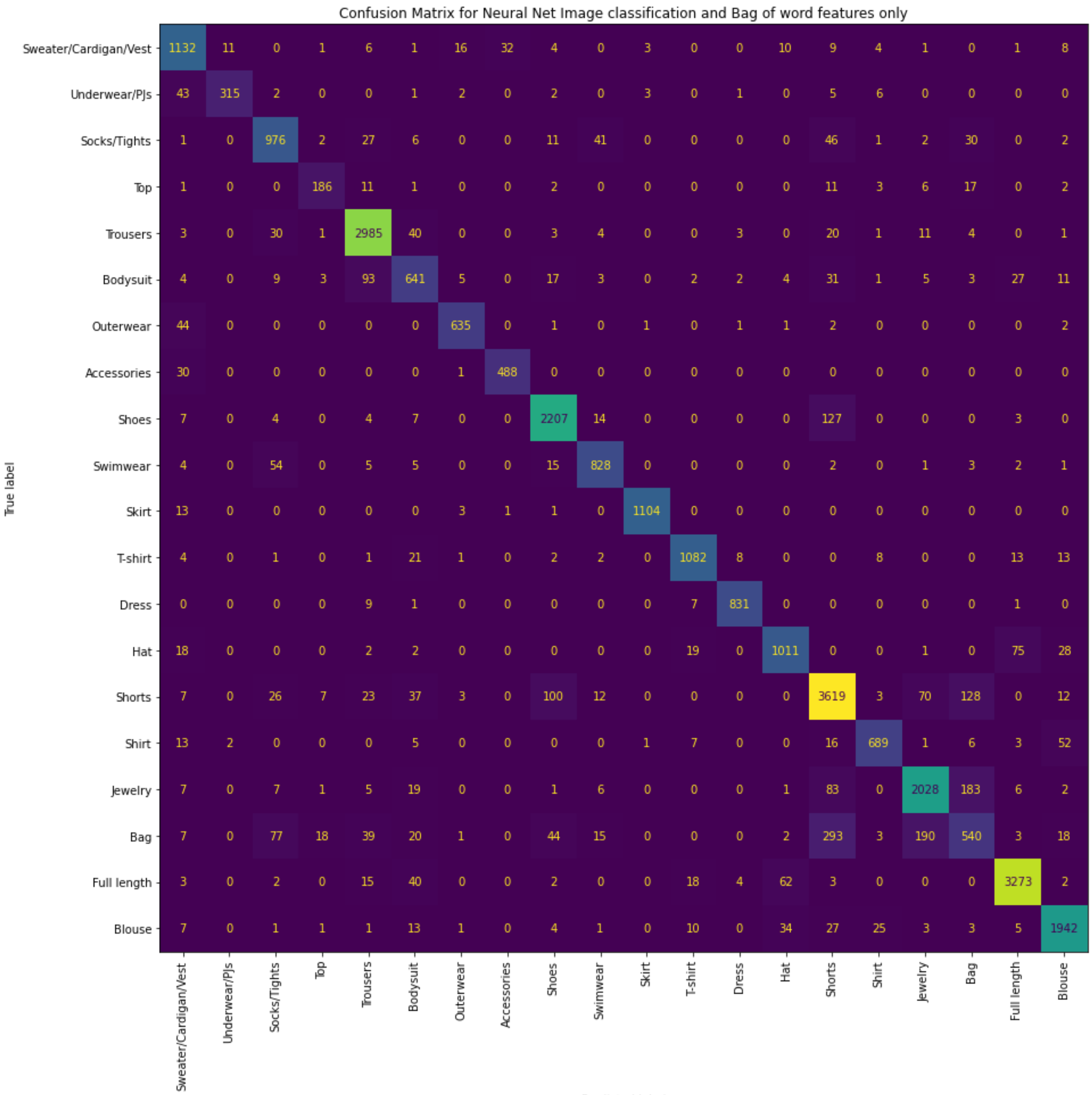
The KNN model also takes the longest out of the 3 models to run.

6.7 Deep Neural Net using Keras

Use the same training and testing sets with the same image embeddings and product title features, train a deep neural net model using Keras.

Steps involved in tuning this 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
- The neural net model had an input dimension of 684 which is the number of feature columns in X_train.
- This neural net model consists of a single hidden layer with 20 output nodes (since there are 20 product categories to predict).
- A dropout rate of 0.25 is chosen to minimize co-adaptation and help reduce model overfitting.
- The crossentrophy metric class is used since this is a multiclass classification problem.
- The weighted average f1-score is used as the metric for minimizing the loss function since this dataset has class imbalances.
- Choose 20% of the training set to be the validation set
- Since neural nets are prone to overfitting, implement early stopping so the model stops training once the validation loss stops decreasing. A patience of 3 epochs is chosen (i.e. training will stop after 3 epochs with no improvement in validation loss)


Confusion Matrix for Neural Net Image classification and Bag of word features only

	Sweater/Cardigan/Vest	Underwear/PJs	Socks/Tights	Top	Trousers	Bodysuit	Outerwear	Accessories	Shoes	Swimwear	Skirt	T-shirt	Dress	Hat	Shorts	Shirt	Jewelry	Bag	Full length	Blouse
Sweater/Cardigan/Vest	1132	11	0	1	6	1	16	32	4	0	3	0	0	10	9	4	1	0	1	8
Underwear/PJs	43	315	2	0	0	1	2	0	2	0	3	0	1	0	5	6	0	0	0	0
Socks/Tights	1	0	976	2	27	6	0	0	11	41	0	0	0	0	46	1	2	30	0	2
Top	1	0	0	186	11	1	0	0	2	0	0	0	0	0	11	3	6	17	0	2
Trousers	3	0	30	1	2985	40	0	0	3	4	0	0	3	0	20	1	11	4	0	1
Bodysuit	4	0	9	3	93	641	5	0	17	3	0	2	2	4	31	1	5	3	27	11
Outerwear	44	0	0	0	0	0	635	0	1	0	1	0	1	1	2	0	0	0	0	2
Accessories	30	0	0	0	0	0	1	488	0	0	0	0	0	0	0	0	0	0	0	0
Shoes	7	0	4	0	4	7	0	0	2207	14	0	0	0	0	127	0	0	0	3	0
Swimwear	4	0	54	0	5	5	0	0	15	828	0	0	0	0	2	0	1	3	2	1
Skirt	13	0	0	0	0	0	3	1	1	0	1104	0	0	0	0	0	0	0	0	0
T-shirt	4	0	1	0	1	21	1	0	2	2	0	1082	8	0	0	8	0	0	13	13
Dress	0	0	0	0	9	1	0	0	0	0	0	7	831	0	0	0	0	0	1	0
Hat	18	0	0	0	2	2	0	0	0	0	0	19	0	1011	0	0	1	0	75	28
Shorts	7	0	26	7	23	37	3	0	100	12	0	0	0	0	3619	3	70	128	0	12
Shirt	13	2	0	0	0	5	0	0	0	0	1	7	0	0	16	689	1	6	3	52
Jewelry	7	0	7	1	5	19	0	0	1	6	0	0	0	1	83	0	2028	183	6	2
Bag	7	0	77	18	39	20	1	0	44	15	0	0	0	2	293	3	190	540	3	18
Full length	3	0	2	0	15	40	0	0	2	0	0	18	4	62	3	0	0	0	3273	2
Blouse	7	0	1	1	1	13	1	0	4	1	0	10	0	34	27	25	3	3	5	1942

Figure 6.7.1 Neural Net Classification Matrix showing the predicted label and the true product category label

The top 3 classes that performed well are: Trousers, T-shirt, Dress
The top 3 classes that performed the worst are: Shorts, Bag, Shoes
This model performs very well and exhibits the same issue as the other models when it comes to handling the problem classes: Bag, Jewelry and Shorts.

6.8 Interpretation of results

Table 6.8 Performance metrics of models on the test set

Model	Weighted f1 score	Test Set Support
Elastic Net Linear model	0.87	29716
Random Forest Classifier	0.85	29716
KNN	0.77	29716
Deep Neural Net using Keras	0.89	29716

Overall, these classification models (Linear, Random Forest, KNN) performed similarly and well enough for the larger 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.

The models even highlighted data that had been misclassified in the original dataset, and I took this opportunity to spot check the rows where the model had the highest predicted confidence scores but the predicted label was different to the true label.

In the colab notebook, I implemented a deep learning architecture using the same features and training and testing sets, to compare the performance of the top 3 and worst 3 classes.

Despite the extra complexity in the model, the neural net performed only slightly better than the linear elastic net model.

7 Applications of the product category prediction models

Managing product data by automating labeling of new products:
An application of these results to solving the business problem of integrating new inventory into the company's existing product taxonomy could be to join these approaches into a multistep classification algorithm where:

- A ML model is used to segment products into a kids vs. adults dataset (this would need to be trained on an existing dataset where labels are known)
- Use the Linear model is used to predict product categories
- Verify the results for product categories where the between product category and product titles is high e.g. 'Dress', 'Shorts', 'Skirt' as shown in Figure 4.2.4

Another approach could be to use a hierarchical taxonomy to predict fewer, broader product categories first instead of treating all 20 with the same weight. Then after the broader product categories are predicted, train additional models for further labeling. For example, a product category 'Accessories' could be used for all Jewelry, Hats, Gloves, Scarves as a way to reduce the class imbalance issue, before training another Accessories ML model to separate the categories into more partitions.

Maintaining correctly labeled data:
The models could be used to flag incorrectly labeled products and inconsistencies in an automated way that can save time and money from having to manually process the data, thus saving operational costs.

Optimizing for sales:
Having products correctly classified in an intuitive, easy to navigate taxonomy helps customers find what they are looking for and boost sales.

Verifying the existing taxonomy makes sense:
Categories that the model gets confused between may indicate categories that should be merged due to high similarity and overlap.

Appendix

The Data:

Column name	Datatype	Description
article_id	int	Index of the dataset. Unique to each product
product_code	int	All products that are color variants are the same product code
prod_name	string	Product title
detail_desc	string	Product description
product_type_name	string	Product category i.e. y label