Proof of Concept

Fashion Trend Intelligence System

Kabir Kumar
Indian Institute of Technology, Delhi

ee3210741@iitd.ac.in, +91-8595671372

Abstract

This is the *Proof of Concept* of my implementation of the assigned project i.e. building a *Fashion Trend Intelligence System* which involves Webscraping, Data Preparation, Feature Engineering, Exploratory Data Analysis, Multi-label Image Classification, Regression and Visualisation

Methodology

The code is implemented in a Google Colab notebook using its GPU hardware accelerator.

Webscraping

- We choose **Myntra** as the host site for our data.
- Rather than the conventional approach of navigating through the website's HTML code using libraries like *BeautifulSoup* & *Requests*, an online extension a.k.a. **Webscraper** is used to perform the aforementioned task.
- A directed graph of selectors is created within the extension window which include Element Clickers (to browse image data containers), Group Selectors (to browse through the specification table), Image Selectors and Text Selectors.
- The choice to use *Webscraper* was taken accounting time constraints and ease of use.

Data Prep. & Feature Engineering

- The data is cleaned using **Pandas** library.
- The cleaning process includes:
  - Dropping useless columns.
  - Removing unrelated words and symbols from relevant columns.
  - Typecasting numeric columns into integer values.
  - Dropping *NaN* value entries.
  - Processing consecutive groups of same entries to extract unique entries.
  - Checking the validity of image URLs and removing invalid entries.
- The feature engineering process includes:
  - Controlling fluctuating values in the price column by normalising it using *Sci-Kit Learn* library in order to improve our models' accuracy.

- Calculating the **Popularity Score** using the normalised prices and no. of ratings to calculate an intermediary called the *satisfaction ratio*.
  popScore = 3 == Affordable, Popular & Satisfactory
  popScore = 2 == Popular & Satisfactory
  popScore = 1 == Less Popular & Less Satisfactory

EDA

The following is plotted using **MatPlotLib** & **Seaborn**:

- Distribution of Popularity Scores
- Distribution of Prices
- Distribution of Number of Ratings
- Popularity Score v/s Normalized Price
- Popularity Score v/s Number of Ratings
- Normalized Price v/s Number of 5 Star Ratings

Multi-Label Image Classification

Processing

- Relevant columns are **one-hot encoded**.
- A dataframe is created for **Style** which includes whether the cloth is a top or not, sleeve length, sleeve style, type & *print (merged into style for easier classification)*.

Computer Vision

Numpy & **Requests** libraries are used.

- Image is opened using the URL and its pixel values are normalised for further processing
- **Data Augmentation** is performed for robust design of model.

Classification Model

Keras is heavily used.

- Transfer Learning is performed on **MobileNetV2** after drawing comparisons with **Resnet50** & **VGG16** in speed and accuracy.
- **Batch Normalisation** layers are added with **ReLU** & **Sigmoid** activation functions (optimal functions for binary encoded data).
- **L2 regularisation** is done to combat over-fitting
- Model is compiled with the **Adam** optimizer, learning rate being **1e-4**, loss function being **categoricalCrossentropy** & metric being **accuracy**.
- The model runs through 10 epochs with batch size of 16.

Regression *(for popularity prediction)*

Processing & Computer Vision

(Similar to the previous method)

Classification Model

Keras is heavily used.

- Transfer Learning is performed on **MobileNetV2** after drawing comparisons with **Resnet50** & **VGG16** in speed and accuracy.
- **Dense** layers are added with the **ReLU** activation function (optimal functions for binary encoded data).
- **Dropout** is done to combat over-fitting
- Model is compiled with the **Adam** optimizer, learning rate being **1e-4** & metric being **MSE**.
- The model runs through 20 epochs with batch size of 16.

Results

Webscraping

- The data is extracted is unorganised with repetitions of the same entry with different attributes missing.

- The limit for pagination is 3 pages per batch. Each batch gives us approximately 600 entries out of which nearly **16%** i.e. nearly 100 are unique and error free.

Data Prep. & Feature Engineering

A clean dataframe is obtained, coupled with enhanced features for providing all-encompassing numerical insights for further processes.

EDA

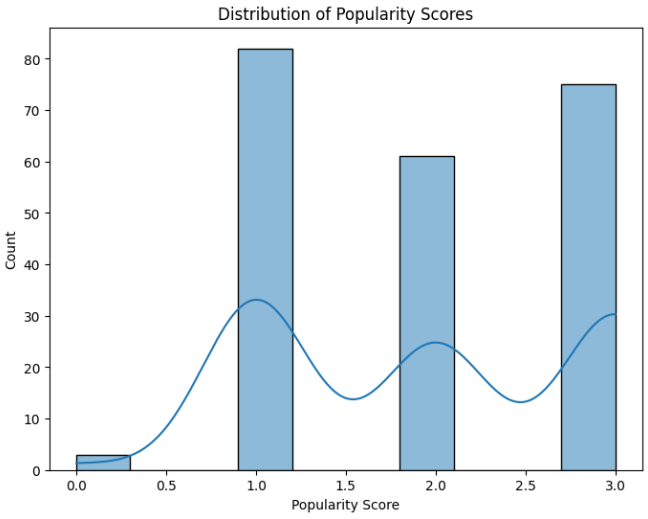

Figure 1: Distribution of Popularity Scores

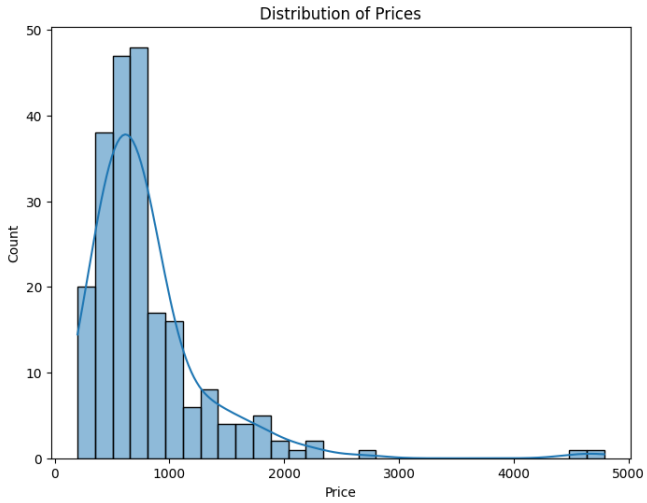

Figure 2: Distribution of Prices

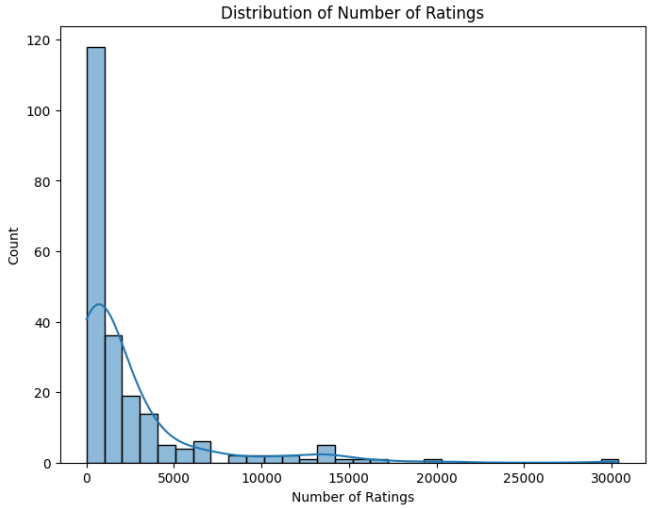

Figure 3: Distribution of Number of Ratings

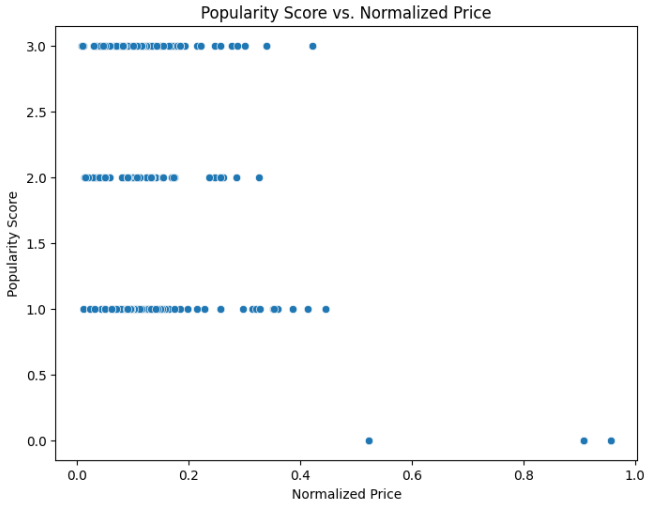

Figure 4: Popularity Score v/s Normalized Price

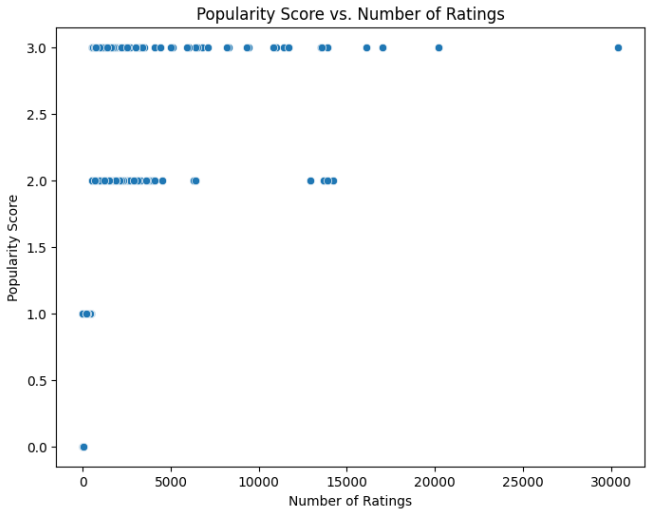

Figure 5: Popularity Score v/s Number of Ratings

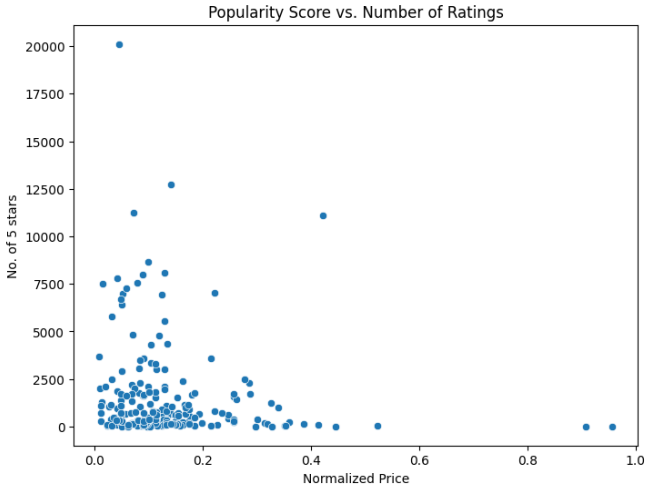

Figure 6: Normalised Price v/s Number of 5 Star Ratings

Multi-Label Image Classification

Epoch	Tra. Loss & Acc.	Val. Loss & Acc.
1	14.6186 (0.0455)	13.7699 (0.0000)
2	10.7921 (0.2045)	13.2786 (0.0000)
3	8.9521 (0.3068)	12.7264 (0.0222)
4	7.8029 (0.4432)	12.4140 (0.0444)
5	7.1389 (0.3864)	12.3971 (0.1111)
6	6.6668 (0.3864)	12.6944 (0.1556)
7	7.0024 (0.3864)	13.0479 (0.1556)
8	6.6587 (0.3580)	13.2654 (0.2000)
9	6.8613 (0.3977)	13.8340 (0.2444)
10	6.5684 (0.3920)	15.0506 (0.2222)

Table 1: Accuracy/Loss (Classification)

Regression *(for popularity prediction)*

Mean Squared Error (MSE) : 0.7867
R-squared (R^2) : 0.0979

Epoch	Training Loss	Validation Loss
1	3.5054	2.5981
2	1.4719	1.4836
3	0.9838	1.7765
4	0.8753	0.6638
5	0.9058	2.1514
6	0.7662	1.2240
7	0.7670	1.4910
8	0.6801	1.1346
9	0.6599	1.8800
10	0.6991	1.1744
11	0.7684	1.7668
12	0.7455	2.3000
13	0.5209	0.9699
14	0.6115	2.1281
15	0.5250	1.8156
16	0.5505	1.9521
17	0.4966	1.7280
18	0.5152	1.9268
19	0.5359	1.6657
20	0.5079	1.4534

Table 2: Losses v/s Epoch (Regression)

Conclusions

This report addresses the challenge of training loss reduction and validation loss increase in multi-label image classification and regression tasks, attributing it to limited high-quality training data. The study serves as a **proof of concept**, demonstrating that by improving the dataset, applying techniques like batch normalization, dropout regularization, and dynamic learning rate scheduling, a more robust model can be built without overfitting.

While Instagram webscraping was omitted from my implementation of this project due to time constraints, the regression model developed offers an alternate solution for predicting popularity using a different dataset.

Figure 7: Samples w/ Actual/Predicted Labels (Classification)

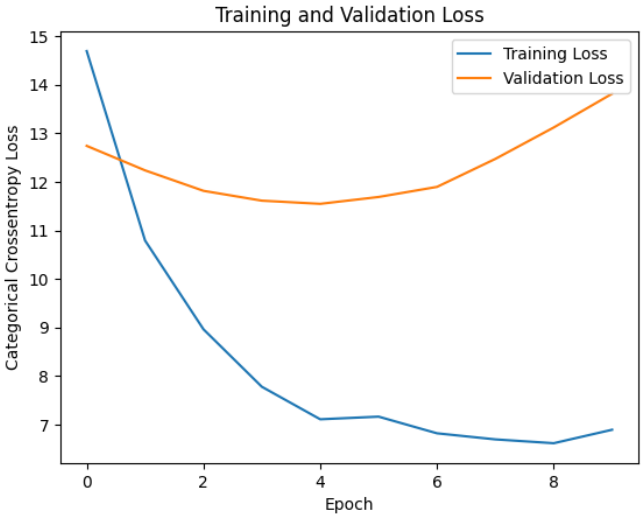

Figure 8: Loss v/s Epoch (Classification)

Figure 9: Samples w/ Actual/Predicted Score (Regression)

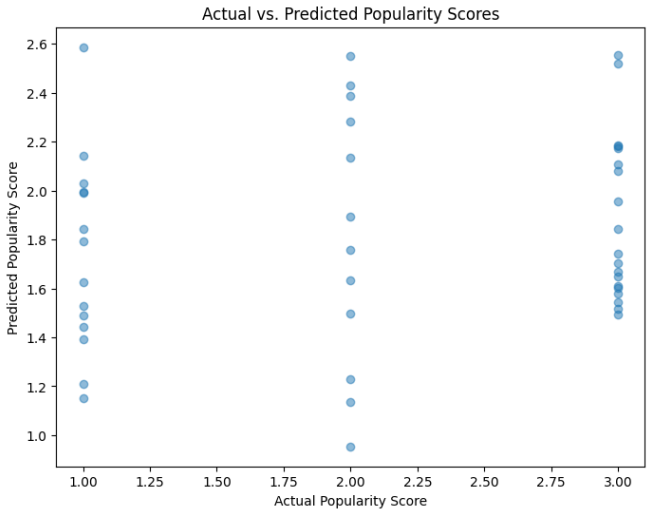

Figure 10: Actual v/s Predicted Popularity Scores

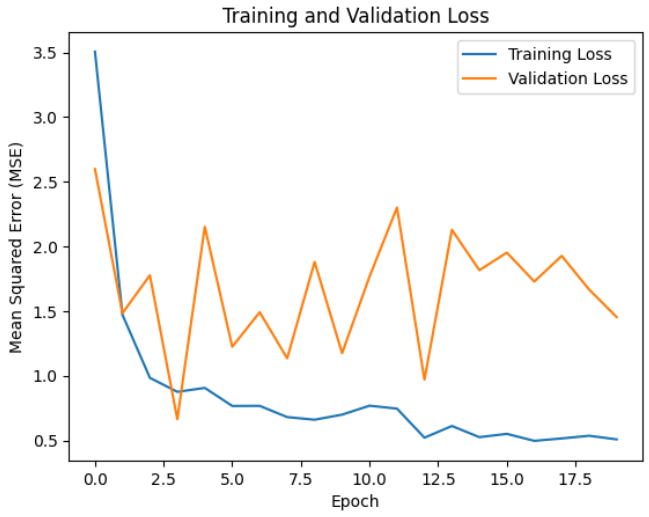

Figure 11: Loss v/s Epochs (Regression)

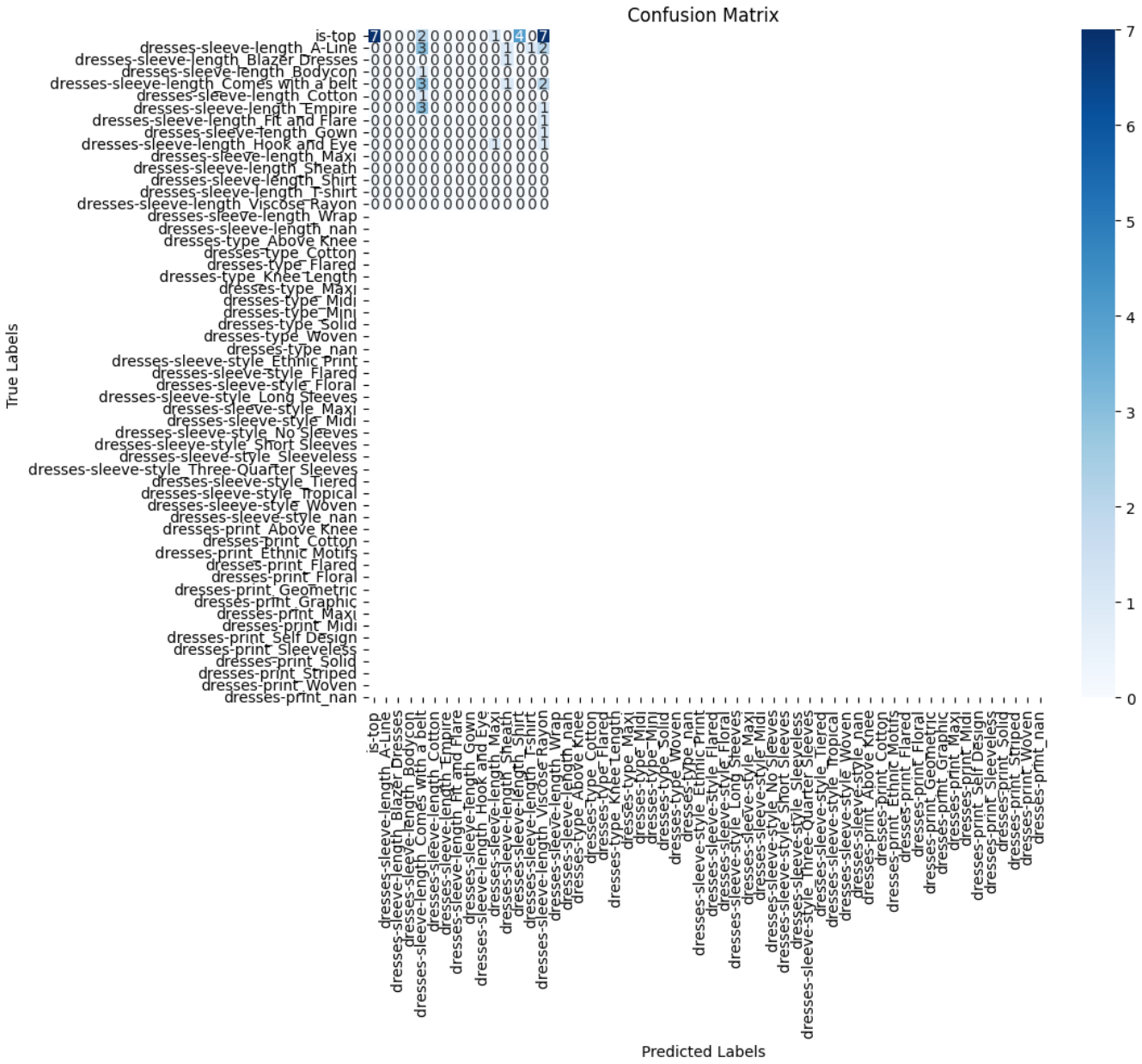

Figure 12: Confusion Matrix (Classification)