Review Classifier Project

Submitted by:

Ashish Kumar Dewangan

1. Project Overview

We have a client who has a website where people write different reviews for technical products. Now they are adding a new feature to their website i.e. The reviewer will have to add stars(rating) as well with the review. The rating is out 5 stars and it only has 5 options available 1 star, 2 stars, 3 stars, 4 stars, 5 stars. Now they want to predict ratings for the reviews which were written in the past and they don't have a rating. So, we have to build an application which can predict the rating by seeing the review.

2. Problem Statement

We have to scrape the reviews fore products like Laptops, Phones, Headphones, Smart watches, Professional Cameras, Printers, Monitors, Home Theater, Router from different e-commerce websites. After collecting the data, we need to build a machine learning model. Before model building we have to do all data preprocessing steps involving NLP. and then try different models with different hyper parameters and select the best model.

The data has reviews and heading of the reviews posted by a user and our task is to classify the the reviews from 1 star rating to 5 star rating. Hence, this problem falls under category of Multi Label Classification Problem.

Multiple ways are there to approach this classification problem which are as follows:

- Multi-label methods which belong to the problem transformation category: Label Power Set (LP), Binary Relevance (BR) and classifier chain.
- Base and adapted algorithms: Naïve Bayes, k-Nearest-Neighbor (KNN).

Further, out of the total dataset used for experimenting these algorithms, 75% was used for training and 25% was used for testing. Each testing dataset was labelled and thus for each algorithm using the predictions and labels, calculation of metric such as hamming-loss, accuracy and log-loss was done. The final results have been complied on the basis of values obtained by algorithmic models in hamming-loss and log-loss combined.

3. Data Exploration

The dataset consists of the following fields-

Heading	It includes the heading of the review in text format.
Review	It is a column with binary values depicting which comments are malignant in nature.
Rating	Binary column with labels for highly malignant text.

Both the heading and review will be preprocessed and fitted separately into different classifiers to predict its rating. Fitting heading will also help us understand if just the headings are sufficient enough for rating classification. The reviews for different types product like Laptops, Phones, Headphones, Smart watches, Professional Cameras, Printers, Monitors, Home Theater and Router will be fitted separately. In addition we will group all the reviews into once category and try to classify them. This will help us generalise the reviews

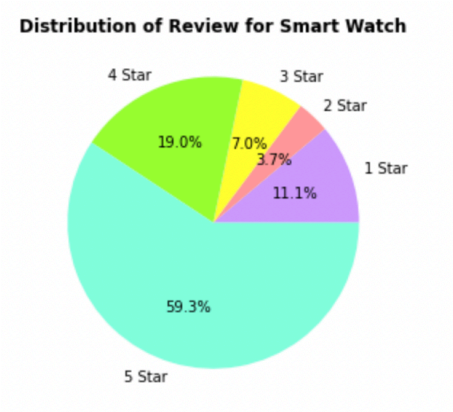
for any product. During extraction all the ratings were entered under one column. But for multi-class classification we need to separate the ratings. The number of unique reviews for each category of products is given in the table below:

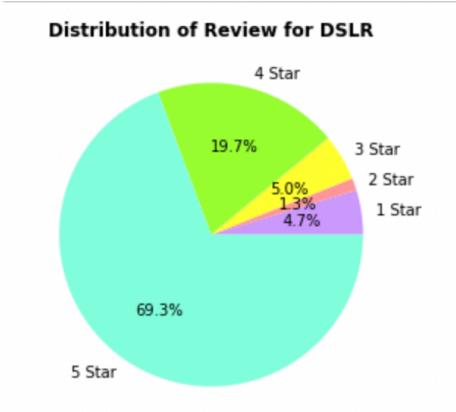
4. Exploratory Visualisation

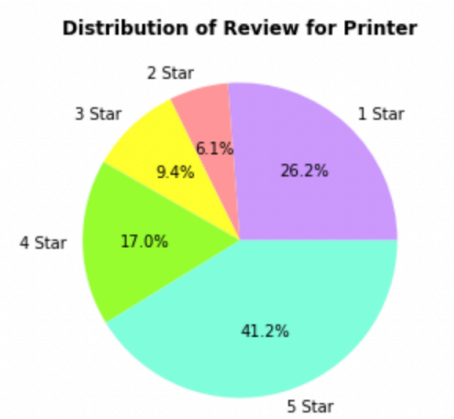
- **Average Length of Reviews -** The average length of reviews varied from product to product. But whole data set had few things in common:

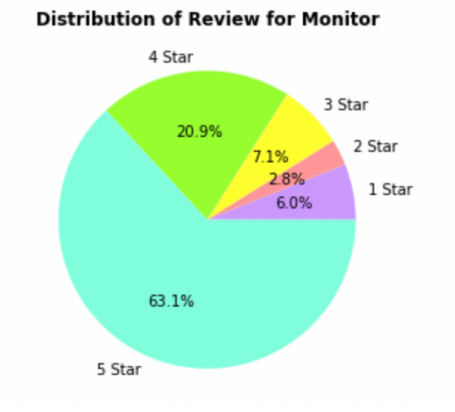
    1. Most of the reviews of the products are rated 5 stars
    2. The length of reviews were less than 600 words.
    3. The length of majority of the reviews was less than 200 words.

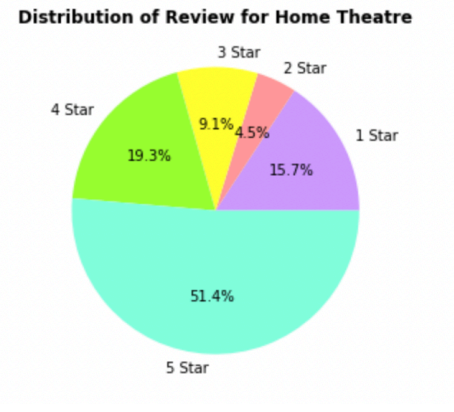
- Distribution of Ratings - For all the products around 50% of reviews were rated 5 stars and around around 5% of the reviews were rated 2 stars. When all the reviews were considered 1, 4 and 5 stars shared similar proportion of the rating 24.6%, 26.3% and 23.7%. At the same time the percentage of 2 star is the least with 8.3%.

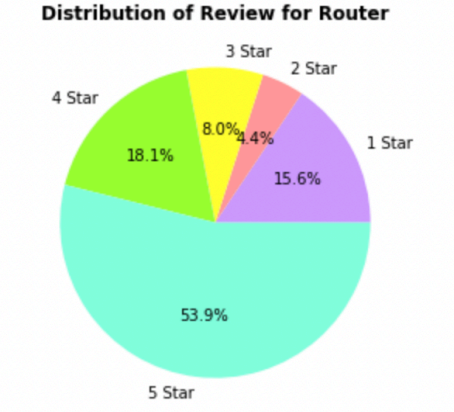

Distribution of Review for Smart Watch
4 Star
3 Star
2 Star
1 Star
5 Star
19.0%
7.0%
3.7%
11.1%
59.3%

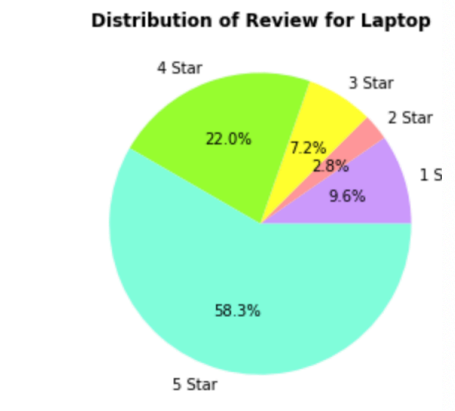

Distribution of Review for DSLR
4 Star
3 Star
2 Star
1 Star
5 Star
19.7%
5.0%
1.3%
4.7%
69.3%

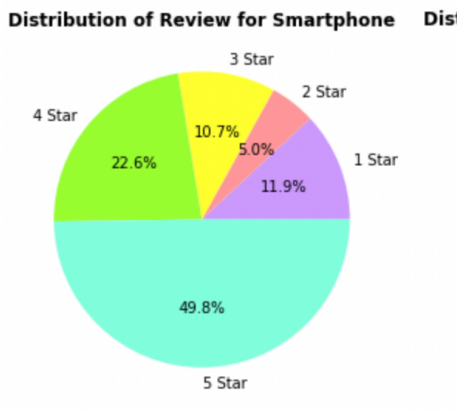

Distribution of Review for Printer
2 Star
3 Star
1 Star
4 Star
5 Star
6.1%
9.4%
26.2%
17.0%
41.2%

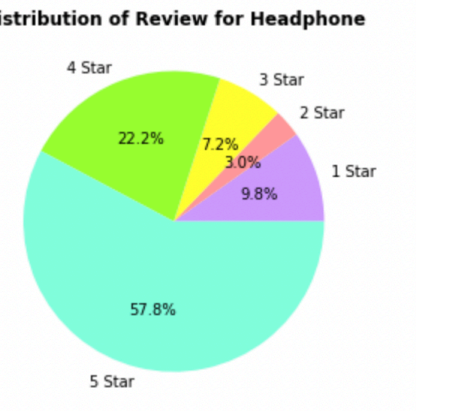

Distribution of Review for Monitor
4 Star
3 Star
2 Star
1 Star
5 Star
20.9%
7.1%
2.8%
6.0%
63.1%

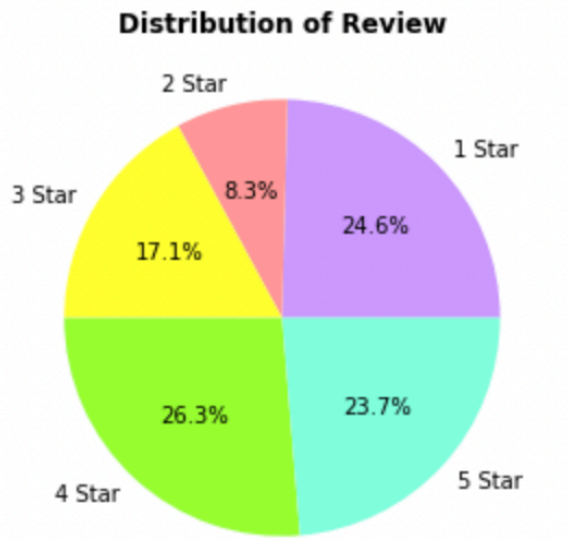

Distribution of Review for Home Theatre
3 Star
2 Star
4 Star
1 Star
5 Star
9.1%
4.5%
19.3%
15.7%
51.4%


Distribution of Review for Router
3 Star
2 Star
4 Star
1 Star
5 Star
8.0%
4.4%
18.1%
15.6%
53.9%


Distribution of Review for Laptop
4 Star
3 Star
2 Star
1 S
5 Star
22.0%
7.2%
2.8%
9.6%
58.3%


Distribution of Review for Smartphone
3 Star
2 Star
4 Star
1 Star
5 Star
10.7%
5.0%
22.6%
11.9%
49.8%


Distribution of Review for Headphone
4 Star
3 Star
2 Star
1 Star
5 Star
22.2%
7.2%
3.0%
9.8%
57.8%


Distribution of Review
2 Star
1 Star
3 Star
8.3%
24.6%
17.1%
26.3%
23.7%
4 Star
5 Star

5. Data Pre-Processing

All the necessary pre processing necessary for NLP was done for both the features heading and review. It was found that the features had emojis in them. So in addition to conversion to lowercase, cleaning of html tags, punctuation removal, stop words removal and stemming, emojis were also removed to get a clean data set.

5. Algorithms & Techniques

As discussed in the Problem Statement section, any multi-label classification can be solved using either Problem Statement methods or Adaptation Algorithms.

Problem Transformation Methods:

- **Binary Relevance Method**: This method does not take into account the interdependence of labels. Each label is solved separately like a single label classification problem. This is the simplest approach to be applied.

- **Classifier Chain Method**: In this method, the first classifier is trained on input data and then each of the next classifier is trained on the input space and previous classifier, and so on. Hence this method takes into account some interdependence between labels and input data. Some classifiers may show dependence such as toxic and severe_toxic. Hence it is a fair deal to use this method.

- **Label Powerset Method**: In this method, we consider all unique combinations of labels possible. Any one particular combination hence serves as a label, converting our multi- label problem to a multi class classification problem. Considering our dataset, many comments are such that they have 0 for false labels all together and many are such that obscene and insult are true together. Hence, this algorithm seems to be a good method to be applied.

Adaptation Algorithms:

- **MLKNN**: This is the adapted multi label version of K-nearest neighbors. Similar to this classification algorithm is the BRkNNaClassifier and BRkNNvClassifier which are based on K-Nearest Neighbors Method. This algorithm proves to give superior performance in some datasets such as yearst gene functional analysis, natural scene classification and automatic web page categorization.

Analytical Problem Framing

1. Implementation

- We will be defining **function evaluate_score** for printing all evaluation metrics: Some implementations return a sparse matrix for the predictions and others return a dense matrix. So, a try except block were used to handle it.

- Then, we will start implementing various **problem transformation methods**. Binary Relevance, Label Powerset and Classifier Chain methods were included.

- **Binary Relevance** method were implemented from scratch. It does not consider the interdependence of labels and basically creates a separate classifier for each of the labels.

- Next, Binary Relevance method for other classifiers (SVC, multinomialNB, gausseanNB) is directly imported from the **Scikit-multilearn** library. Classifiers for **Classifier Chain** and **Label Powerset** are imported and tested.

- Then **Adaptation Algorithms** were used. To be precise, MLkNN will be imported from the scikit-multilearn library.

- The **evaluate_score function** cannot be used to operate directly on the predictions of this model as it returns **probabilities in a range of values from 0 to 1**. The hamming_loss and accuracy_score cannot work on these values directly. Hence, predicted results were rounded.

2. Refinement
Problem Transformation Methods:

- Our initial model was the Binary Relevance method using **MultinomialNB** classifier.

- We then tried other classifiers **GaussianNB**:

- Since MultinomialNB was giving the best results until now, **Classifier Chain** and **Label Powerset** was implemented using this classifier only.

Adaptation Algorithms:
MLkNN was implemented to get further improved results.

3. Results

Free Form Visualisation
The hamming-loss and log-loss of different models used. We plotted in 2 scatter plots:

- If we compare all the models on hamming-loss: The best model will be Classifier Chain with Multinomial NB Classifier.

Product	Hemming Loss				Log Loss				Accuracy			
	CC	LB	BR	MLkNN	CC	LB	BR	MLkNN	CC	LB	BR	MLkNN
Laptop	0.15	0.34	0.57	0.23	9.37	29.44	5.88	12.85	47.12	14.76	6.92	25.07
Smartphone	0.14	0.32	0.50	0.16	9.84	27.70	19.70	11.74	56.47	19.79	13.04	54.01
Headphone	0.14	0.32	0.50	0.16	9.84	27.70	19.70	11.74	56.47	19.79	13.04	54.01
Smart watch	0.14	0.31	0.44	0.15	9.38	26.93	18.58	10.46	58.30	22.02	14.51	56.33
Professional Camera (DSLR)	0.12	0.32	0.47	0.12	9.66	27.93	22.03	10.17	68.39	19.12	14.83	64.98
Printer	0.15	0.31	0.53	0.17	7.92	26.99	6.15	11.81	45.16	21.86	10.25	48.17
Monitor	0.14	0.33	0.53	0.15	10.21	28.87	21.15	11.14	61.28	16.41	10.34	55.82
Home Theatre	0.15	0.33	0.51	0.19	9.63	28.49	13.07	13.93	51.12	17.52	8.89	46.78
Router	0.14	0.33	0.55	0.21	9.43	28.42	19.46	14.85	55.47	17.72	9.61	46.63
Overall	0.17	0.27	0.62	0.26	5.60	23.22	4.83	17.66	27.36	32.75	3.85	29.37

- If we compare all the models on log-loss: The best model will be Binary Relevance with Gaussian NB. However log loss of Classifier Chain with Multinomial NB is comparable to Binary Relevance with Gaussian NB
- If the product was considered individually then also CC with Multinomial NB performs better than others.

Thus, Classifier Chain with Multinomial NB is selected as our final model.

Conclusion

- The first step involved collecting data and structuring it to make it suitable for training.
- The second major step was performing cleaning of data including punctuation removal, stop word removal, emoji removal, stemming and lemmatizing. This step was also crucial since the occurrence of similar origin words but having different spellings will intend to give similar classification, but computer cannot recognize this on its own. Hence, this step helped to a large extent in both removing and modifying existing words.
- The third step was choosing models to train on. Since I had a wide variety of models(3 for problem transformation) and classifiers(not bounded) along with number of adaptation models in MLL kNN, selecting which all models to train and test took lots of efforts.
- Finally comparing on the basis of different evaluation metrics. The two major evaluation metrics I planned to compare on were hamming-loss and log-loss. Hence the final model selection was done on the basis of the combination of both these losses.