Reviews Classification using Neural Network
Implementation of Multi-Layer Perceptron (MLP)

Drishte Assignment Task Submitted by Yashi Kesarwani

Objective and Datasets Information

- **Goal - The Goal of this assignment is to implement a multi-layer perceptron from scratch.**
- **Objective- The objective of this project is to classify the reviews given by users as positive or negative.**

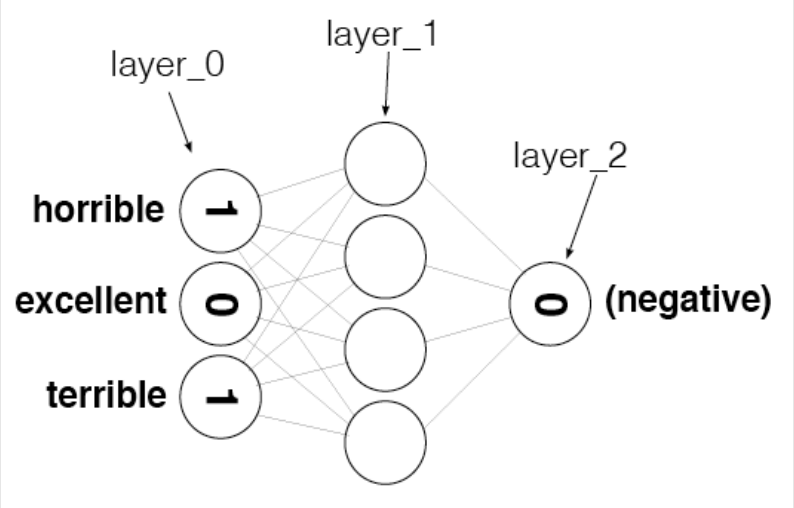
- **Reviews.txt** is the dataset of the collections of reviews given by different users. **Label.txt** is the dataset of the corresponding results of the same row of the reviews.txt dataset i.e. POSITIVE or NEGATIVE.
- **Dimension of the dataset**: Length of reviews and label dataset is 25000. Each row of label dataset gives result to the corresponding review result of reviews dataset.

Approach

- First of all we are curating the dataset, We create three Counter objects, one for words from positive reviews, one for words from negative reviews, and one for all the words.
- Calculated x = positive to negative ratio for each word using **positive_counts[word] / float(negative_counts[word]+1)**.
- If x > 1, word is considered as positive word and if x < 1, word is considered as negative words.
- Convert all the ratios to logarithms. (i.e. use np.log(ratio))
- Transforming Text into Numbers
- Creating a set named vocab that contains every word in the vocabulary. The length of this vocab is 74074.
- **update_input_layer** is counting how many times each word is used in the given review
- **get_target_for_labels** is returning 0 or 1, depending on whether the given label is NEGATIVE or POSITIVE, respectively.

Approach (tbc..)

- Take a look at the following image. It represents the layers of the neural network we'll be building throughout this project. layer_0 is the input layer, layer_1 is a hidden layer, and layer_2 is the output layer.



- Creating a numpy array called `layer_0` and initialize it to all zeros. Here we create `layer_0` as a 2-dimensional matrix with 1 row and `vocab_size` columns.

Approach (tbc..)

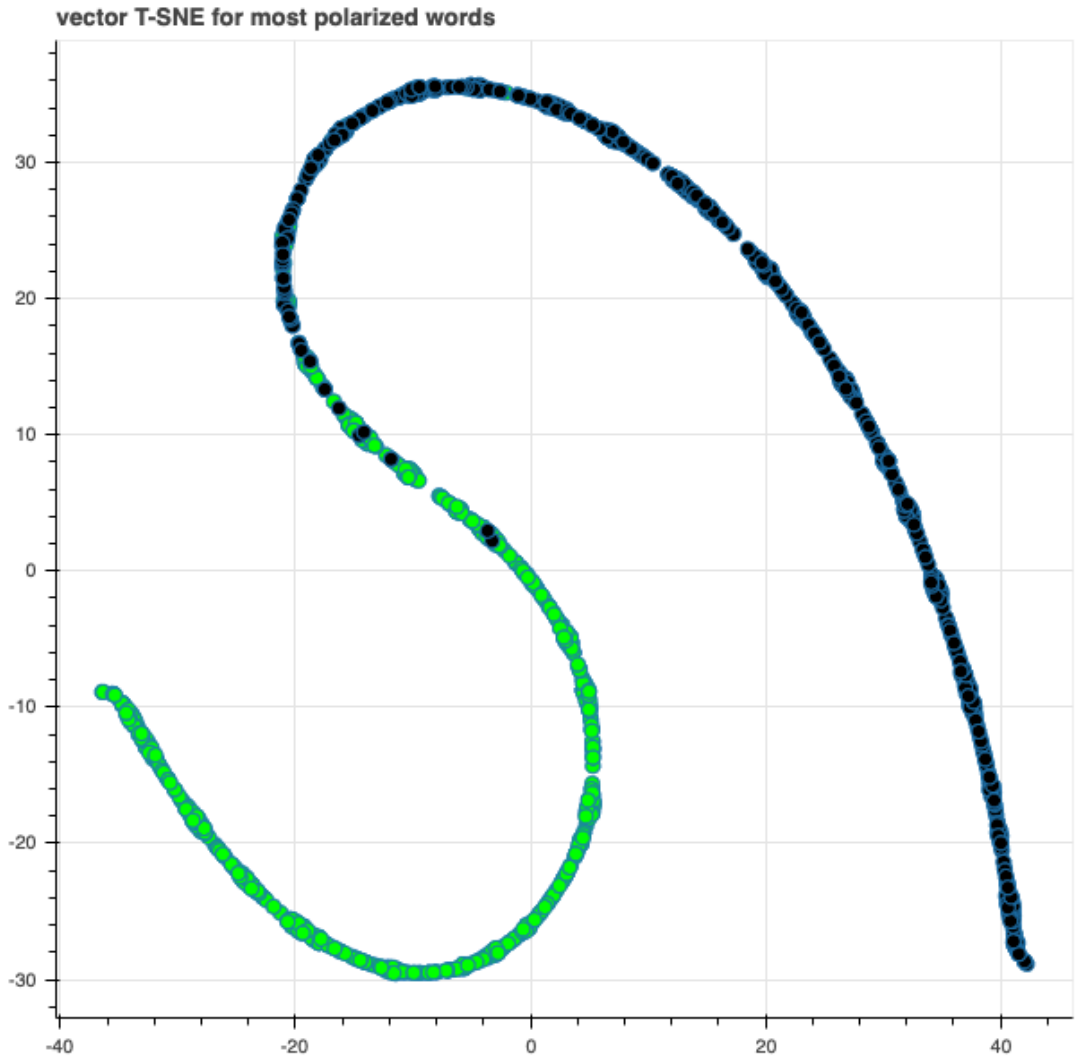
- Building a Neural Network - Creating a Sentiment Network with the given parameters : **reviews(list)** - List of reviews used for training, **labels(list)** - List of POSITIVE/NEGATIVE labels associated with the given reviews, **hidden_nodes(int)** - Number of nodes to be created in the hidden layer, **learning_rate(float)** - Learning rate to use while training.
- We used Backpropagation algorithm and performed forward pass and backward pass to initialize and update weights and errors of the layers.
- Then training and testing the model using dynamic learning rates.
- Detecting different noise and tried to remove it from model to improve the accuracy of the model using additional parameters like **"min_count"** and **"polarity_cutoff"** and also used dynamic polarity_cutoff rate.
- Analyzing the inefficiencies and make the model more efficient by removing the inefficiencies and training the model again.

Final accuracy

- The final training accuracy of the model is 85.6 %
- The final testing accuracy of the model is 85.9 %

Visualization

- **Visualization of Word Vectors as Positive or Negative -** green indicates positive affecting words, black indicates negative affecting words



Application

- This model can be used to analyze whether a given product or anything in the market like a movie, any new startups, etc. has more positive influence or negative influence.

THANK YOU!