PSL Project 3 Report

Sushanta Panda (net Id = panda5)

Goal

Goal of the project is to predict the “Sentiment” against the review text captured against each movie id.

Input & Output

We need to build the model, where the input to the model will be “review” (as a vocab matrix) and sentiments (0 or 1) and the “myvocab.txt” (which will be used as a master vocab list to converts the train data’s review based on this “myvocab.txt”) in order to train the model. Once the model is trained, when the test data will be input (“review” data will be converted as per “myvocab.txt”), the output will be “sentiment” (0 or 1) against the “movie id”.

Vocabulary Size

Below are the steps carried out to get a Vocabulary Size of 1900

- **Remove HTML tag <br /><br />**– Remove the HTML break tag from the “review” text

- **Convert the text into TfIdfVectorized** – Convert the review of the text into TfIdfVectorized, which also includes to exclude the stopwords (['the','with','he','she','also','made','had','out','in','his','hers','there','was','then'])

- Use the “ttest_ind” from the scipy packages to calculate the T-test for the means of two independent samples

    - Generate & Store the tstat score against each word retrieved from the TfIdfVectorized package
    - Generate & Store the tstat score with absolute value for each word retrieved from the above code
    - Sort the value based on the magn_tstat in descending order
    - Pick the first 1999 vocab from the list based on the magn_tstat values

Fitting Prediction Model

Below is the single model created which gives the AUC score above 0.96

1. **Logistic Regression**
    a. Logistic Regrssion seems only model which runs faster as compared to another model (Xgboost and RandomForest, runs over 1 hr and the ROC AUC score is not that great).
    b. Initially runs the Logistic Regression with alpha lower value (0.09, 0.1), where the AUC score comes as 0.93, however for alpha = 17, the AUC score improved to above 0.96
    c. The default penalty used as "l2"
    d. As part of the model, did the following
        i. Load the "myvocab.txt" and train.tsv and test.tsv data
        ii. Based on the vocabulary list from "myvocab.txt", transformed the vocabulary lists from the train and test into array of matrix for which only vocab from the "myvocab.txt" will be retained (which will be 1999 words).
        iii. Then the array would be feed into the Logistic Regression with Alpha = 17 with "l2" penalty to evaluate for AUC score, which comes as > 0.96 for all the 5 splits

Interpretability

As part of the Project, I am using the "Logistic Regression" model to predict the "sentiment" against the movie's review. This is how the model works

1. **Feature vector**: Before the model takes the data as input, created the feature vector where each feature is a word and there is a number (datatype = float) assigned against the feature (i.e. word) which comes from the "review" against the "movie id". The value against the feature (i.e. word) is either "0" (if the specific word is not present in the "review" text against "myvocab.txt") or have the specific float number comes from the "TFidf Vectorize".

2. As part of the project, we need to take the vocab size <= 1000 and see the AUC >= 0.96, I have managed to take the total vocab size around 1900 for which the AUC is above 0.96.

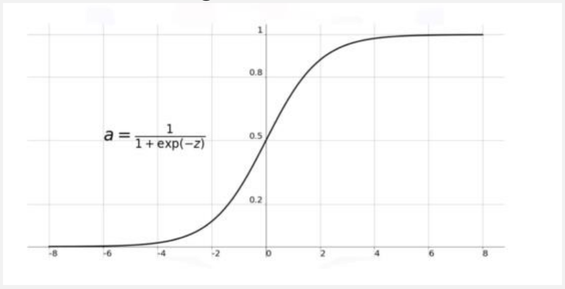
3. These 1999 words which converts into 1999 features. So, as part of the "Logistic Regression", we will use the "Sigmoid Function" which is the main "objective function" (mentioned below), to calculate or derive the "sentiment" against a "movie id" based on the "reviews". So, what we are interested to know the co-efficient (w1, w2, w3, …, w1997, w1998, w1999) against each feature (i.e., word). This is because, based on the co-efficient, we will multiply with the value of the feature vector to derive the value of "X" (as mentioned below), which is then plug-in into the sigmoid function to derive the value of "y" or S(x) which is the "Sentiment" against the "movie id"

X = w0 + w1*(word1) + w2*(word2) + w3*(word3)+ ..... + w1997*(word1997)+ w1998*(word1998) + w1999* (word1999)

Sigmoid Function

$$S(x) = \frac{1}{1+e^{-x}}$$

Sigmoid Function



4. Now we the feature (each word which I think is valuable as part of a feature and 1999 words) and the function (sigmoid) for which we can calculate the “sentiment” for any new movie’s review. Now, the only thing is how to calculate the 1999 co-efficient and 1 intercept. For any “object function”, if we have a cost function (especially if it’s a convex), we can take the derivate (or calculation of slope) to calculate these co-efficient. So, what would be the cost function? Below is the “cost function” we can have for the “Logistic Regression” and it’s a convex one

$$-\frac{1}{m}[\sum_{i=1}^{m} y^{(i)} \log(h_\theta(x^{(i)})) + (1-y^{(i)}) \log(1-h_\theta(x^{(i)}))]$$

Where

- h(x) = Sigmoid function = We can plug-in the value from the above formula
- m = number of samples

5. Then, we need to take the derivative of the “cost function”, based on x(i) with respect to w0 (intercept) and co-efficient (w1, w2, ….,w1997, w1998, w1999)

6. From all the above, it’s pretty clear that, the derivation of the “Logistic Regression” is purely depends upon what feature set we have and obviously during the training of the model if certain words in the “review” define the “sentiment” of the movie, it will also have the similar effect on deciding the “sentiment” for a new movie.

Logistic Regression’s AUC score and Time Taken

Split	AUC Score	Time Taken (In Seconds)
1	0.9634	34.2
2	0.9617	33.8
3	0.9622	34.7
4	0.9639	34.3
5	0.9619	33.3

Hardware Configuration

- iMac Pro (2017) 3.2 GHz Intel Xeon W, 32 GB 2666 MHz DDR4
- Jupyter Notebook 6.0.3, Python 3.7.8

References

- **Kaggle** - https://www.kaggle.com/lakshmi25npathi/sentiment-analysis-of-imdb-movie-reviews