University College London

Information Retrieval and Data Mining
COMPGI15/COMPM052

2018 Course Project: News Stance Detection

Jenny Chi Yan Yang
ID: 17114792

Introduction

As information is getting overwhelming on internet, article writers may use attention catching headlines to attract readers, that sometimes these headlines may be misleading carrying different message than the article they are representing. That's how the study in fake news and news stance become so popular and important in data mining and natural language processing. In this assignment, I worked on a data set with news headline and bodies to analyze their similarities and to further create models to estimate their stance.

Datasets (Data Splits and Statistics)

The data set was heavily imbalanced between the 4 stances, for later model training purpose I have converted the stances into number where 1: Agree, 2: Disagree, 3: Discuss, 4: Unrelated. The article bodies were first merged to the stance data frame by the body ID and then the train set is further split in to training set and validation set with a ratio 9:1. The ratio among the 4 classes have been reserved across 3 data sets.

	Stance	S
0	agree	1.0
1	disagree	2.0
2	discuss	3.0
3	unrelated	4.0

Training set - Real Stance count

	S	Stance	Percent%
0	1.0	3310	7.359808
1	2.0	756	1.680971
2	3.0	8018	17.828078
3	4.0	32890	73.131142

Validation set - Real Stance count

	S	Stance	Percent%
0	1.0	368	7.362945
1	2.0	84	1.680672
2	3.0	891	17.827131
3	4.0	3655	73.129252

Test set - Real Stance count

	S	Stance	Percent%
0	1.0	1903	7.500985
1	2.0	697	2.747339
2	3.0	4448	17.532519
3	4.0	18322	72.219156

Figure 1: Datasets Statistics

Preprocessing

To do sematic analysis, the text must be preprocessed otherwise the accuracy would be very low. Below are several cleaning steps that I have performed:

1. Tokenized the sentences into individual words with
2. Filtered out punctuations, number, and other non-English symbols
3. Changed all upper-case alphabets to lower-case
4. Converted plural words to singular words
5. Cross checked with the English stop word dictionary in nltk library to remove stop words
6. Filtered an unique vocab list saved separately for model training later

After these 7 steps, the list of useable words was saved in a new column in the data frame, ready for further calculation and analysis. Same process was done on all 3 data sets.

Vector based representation

For vector representation, I have applied Word2Vec method from gensim library which is a shallow neural network model with single hidden layer. Word2vec has no activation function on the hidden layer neuron but a uses a softmax function for classification on the output layer. This is more efficient than the bag of word approach because instead of creating spare one hot vectors with dimension of the entire population of vocabulary in the corpus, the features can be encoded into dense embedding matrix which in isolation carry no semantic meaning interpretable by human. In my model, I have used the default size of 100. To train the model efficiently, I have used a unique vocabulary list and then I saved the learned word vectors to a dictionary, named 'vocab' in my program.

Cosine Similarities

With the vector representations extracted with the function I built, 'vectorizer' in the program, I could use them to calculate the cosine similarities to measure the similarity between the headlines and article bodies, defined between cos(0)to cos(90), thus scaled between 0 and 1 with the closer to 0 the more similar and 1 meaning completely different. It is a most commonly used distance metrics as it has removed

the influence of the word frequency, vector length, as the theme of a document should be independent from its length. While calculating for the test set, there were 43 headlines containing new vocabularies not in the trained word2vec model. Those were given 0 cosine similarity for filtering out later, because there are no pair with 0 cosine similarity in our datasets.

$$similarity = \cos(\theta) = \frac{A \cdot B}{||A||\,||B||}$$

Language Model based representations

The word2vec model that I used above was actually a skip-gram model which maximizes the context word probability by minimizing the negative log likelihood with the given center word to predict its surrounding words. The window size is randomly selected by the neural networking within the predefine maximum size.

Additionally, I have established a unigram-based representation which belongs to the family of n-gram models but with a size of 1. It calculates the probability of the word, similar to the bag of words model. I first calculated the frequency of each unique word and calculated their probabilities which some to 1 in total. The probability of each headline or body will be the product of the probability of all the words in the corpus.

Another model of representation is Divergence Model, whose base theory is the Kullback-Leiblar (KL) divergence which measures the cross-entropy between 2 probability distribution. With the probabilities that I have calculated for the headlines and bodies, I then followed the below equation to calculate their KL divergence to compare their similarities.

$$\text{KL divergence: } H(M_Q||M_D) = -\sum_{w} p(w|M_Q) log\ P(w|M_D)$$

To train a model that would be applicable to unseen documents, missing word in the dictionary need to be addressed. This can be resolved by the smoothing techniques. The basic Laplace smoothing has been applied in my model. To obtained renormalized probability with Laplace estimates, I added 1 to the numerator and total number of words to the denominator.

$$\text{Laplace estimates: } (\frac{m_1 + 1}{|\text{D}| + |\text{V}|}, \frac{m_{21} + 1}{|\text{D}| + |\text{V}|}, \dots\dots, \frac{m_{|v|} + 1}{|\text{D}| + |\text{V}|})$$

Alternative Distances and feature selections

I have explored and computed the Manhattan Distance and Euclidean Distance for comparing text similarities. I have chosen these 2 distances because Euclidean is the feature that people compare with cosine similarities the most. Euclidean distance is the shortest straight-line distance between 2 points on the Euclidean space. In context of sematic analysis, it is measuring the scalar distance between 2 vector representations. In the paper [2], author proposed to use term frequency and the length of vector in measuring word significance in documents. I think it can also be applicable in news stance detection as the higher a word's significance in the article bodies, the high probability it appears in the headlines.

$$\text{Euclidean distance} = d(p,q) = d(q,p) = \sqrt{\sum_{i=1}^{N}(p_i - q_i)^2}$$

Manhattan Distance calculates the shortest distance between 2 points on a grid. By definition it was calculated as the sum of 2 points' absolute X and Y Cartesian coordinates difference. Although it is not

commonly applied in natural language processing, according to this paper [2], in measuring distance in high dimensional space, as the dimension increases the meaningfulness of the measurements drops as k increase in L_k norms, especially in L_3 or above. Authors suggested that Manhattan distance L_1 is the most suitable for calculating high dimensional distance, second best will be Euclidean distance L_2.

$$\text{Manhattan distance} = d(p,q) = ||p-q||_1 = \sum_{i=1}^{n} |p_i - q_i|$$

Eventually I have chosen to use cosine similarity, Euclidean distance and Manhattan distance to be my training features for my regression models. My selection is further supported by the distribution graphs of the 4 different types of measurements. In the first 3 graphs in figure 2, we can see the median and the whole distribution of the unrelated stance are significantly shifted away from the other 3 stances, also the kurtosis of the disagree are also slightly higher. However, presenting in KL divergence, all 4 stances have highly similar shape of distributions which indicate it is not a good feature for the model to learn different classes' characteristics. Being plotted in a 3D space, the selected 3 features have presented an observable trend in their clustering, proven to be suitable for a regression model.

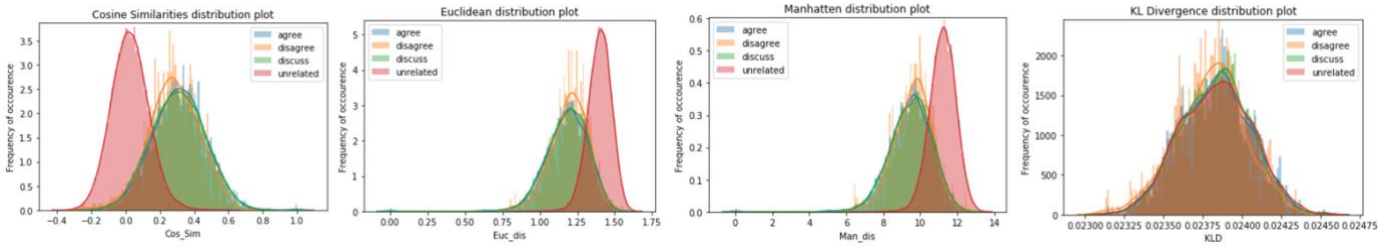

Figure 2: 4 Distances Distributions of 4 Stances

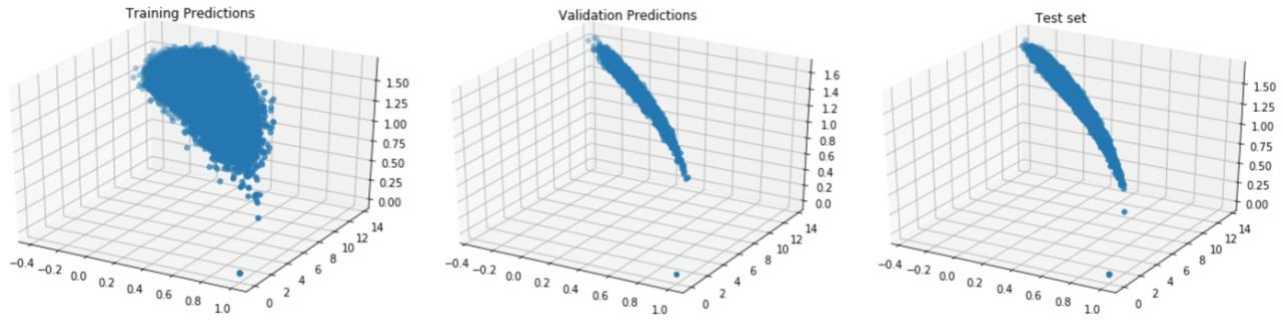

Figure 3. 3 Training Features of 3 Datasets 3D Scatter Plots

Models

Detecting news stances is a multi-class classification problem. As traditionally linear regression is used for continues value prediction and logistic regression is for binary classification, there are 2 ways to twist them to fit into a multi-class classification. Method 1 is to round the value prediction results to the nearest integers. Method 2 is to do 4 times of one verse all classification and combine the results. I have applied method 1 for linear regression and method 2 for logistic regression in this exercise. In both of the linear regression model and logistic regression model, I have used gradient descent in finding the minimum of the cost function.

For the evaluation metrics, I have used root mean square error and R2 score to examine the training and testing results. Root mean square error (RMSE) measure the predictive power of a regression, however, it doesn't have standard range for reference as the value size depends on the unit of the predicting variable y. To make it more meaningful I normalized it by dividing it by the y mean to form NEMSE. R^2 , the coefficient of determination, measures a function's goodness of fit to the data by calculate the ratio of explained variation and total variation. It ranges between 0-100% with generally the higher the better fit.

SSres is the regression sum of square and SStot is the total sum of square. In our case both measure how well the regression models predict comparing against the true values in datasets.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \qquad NRMSE = \frac{\sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}}{\hat{y}} \qquad R^2 = \frac{SS_{res}}{SS_{tot}}$$

Linear Regression

For linear model, the results are better than logistic regression. From the histograms in figure 4, we can notice some news got misclassified as the 5th class, thus the total percentage of 4 classes does not sum to 100. On Testing set, it has classified more to disagree and discuss, less to agree and unrelated, comparing to the true ratio in figure 1. The RMSE is 0.75 slightly higher than in training and R2 score is 0.26, lower than in training. NRMSE are consistently in the 0.19-0.21 range.

The learning rate (α) is 0.01 optimal with minimum cost 0.22 on training set. Learning rate determines the step size we update the weight after each run and getting the new gradient. Taking too big steps might cause missing the minimum and too small steps may cause a long time or never get to converge. The results from linear model were significant improved after tuning α that RMSE was lowered to 0.68 from 0.72.

$$Gradient\ decent: (repeat\ until\ convergence)$$

$$\theta_0 = \theta_0 - \frac{\alpha}{N}\sum_{i=1}^{N}\left(h\left(\theta^T x^{(i)}\right) - y^{(i)}\right) \qquad \theta_1 = \theta_1 - \frac{\alpha}{N}\sum_{i=1}^{N}\left(h\left(\theta^T x^{(i)}\right) - y^{(i)}\right)\cdot x^{(i)}$$

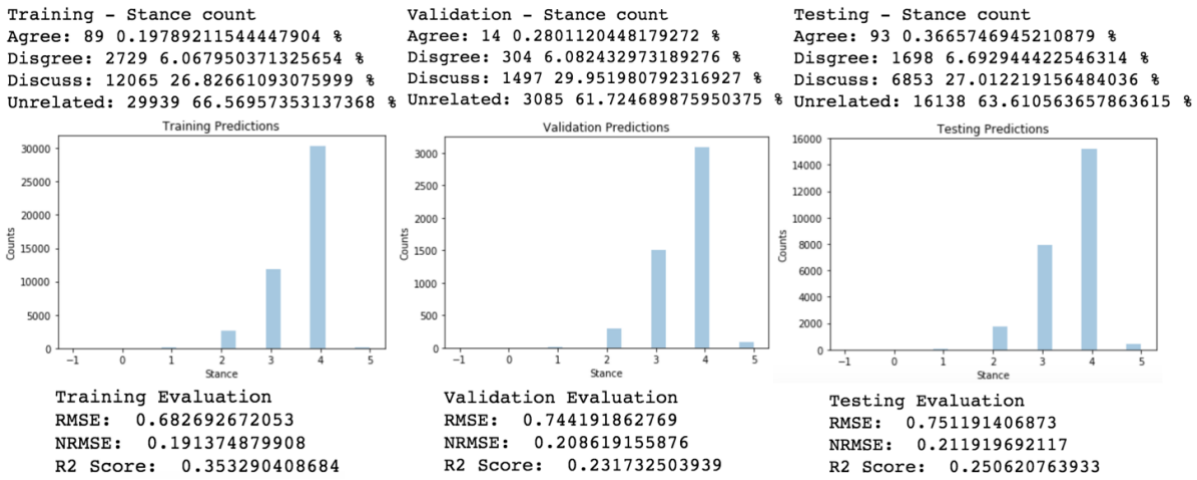

Figure 4. Linear Regression Results

Logistic Regression

Logistic regression is best for 2-class independent categorical classification problem, and the activation function is sigmoid. The learning rate which is 0.000001, much lower than linear regression. The accuracy of the prediction results was also lower which is 0.80 on training set and 0.84 on testing set, with only slight improvement after tuning learning rate. Minimum cost 0.31 was also higher than linear models' 0.22. Overall the logistic regression model has underperformed the linear model.

$$Sigmoid\ function: \quad S(x) = \frac{1}{1 + e^{-x}}$$

$$Minimizing\ lost\ function: \quad \frac{\partial h(\theta)}{\partial \theta} = -\frac{\partial}{\partial \theta}\sum_{i=1}^{N}\left(y^{(i)} log(h(\theta^T x^{(i)})) + (1-y^{(i)})\log(1-h(\theta^T x^{(i)}))\right)$$

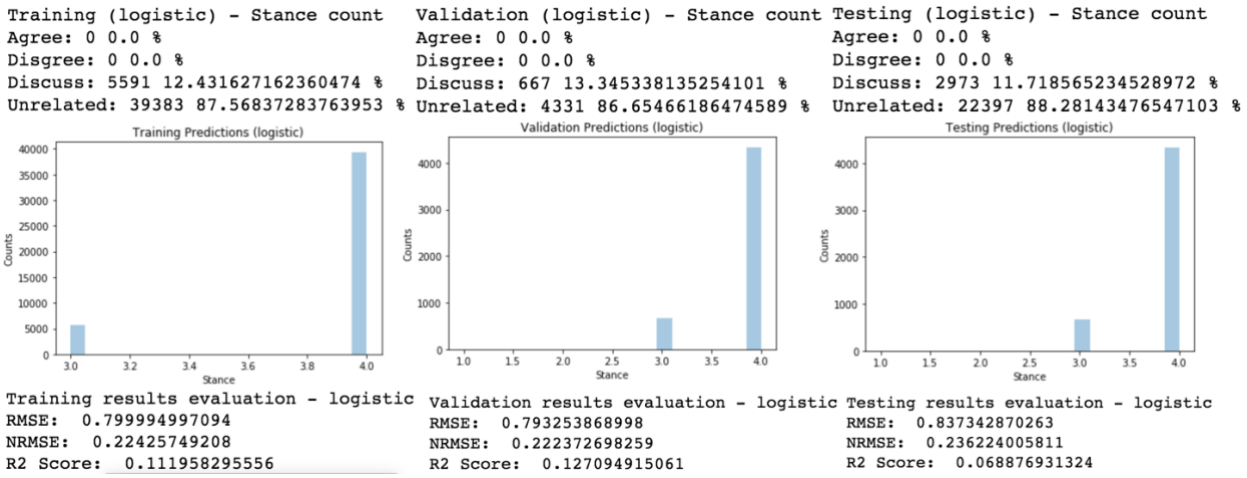

Figure 5. Logistic Regression Results

Improvement work

For alternative method, I propose using Xgboost which stands for extreme gradient boosting, base model is decision tree. Tree-base models usually perform very well in multi-class classification problems. I have used the Xgboost package to train the classification model and the prediction results on testing set were better than both linear and logistic regressions after tuning, with accuracy score 0.8143, RSEM 0.7365, NRSEM 0.2078, R2 score 0.2795. For the tuning, I have set max_depth as 3, 100 n_estimators, 0.1 learning_rate, gamma as 0.3, min_child_weight as 9, colsample_bytree as 0.7 and 0.8 subsample. The run time is also just as fast as the previous 2 regressions.

When the results were still not good enough even we picked the right models and after tuning them, we will then search ways to improve the input data. Assuming the data colloction method is appropriate that our sample data does represent the population, due to the imbalance of classes in the original data sets, I suggested to do some rebalancing between the 4 classes in the training set. For example, upsizing the agree, discuss and disagree, downsizing the unrelated. The purpose is to have enough data of each class in the training set allowing the model to pick up characteristics of each class for better leaning.

Literature review and conclusion

From my literature review, I have notice on top of word frequency, the word order is another very important feature to take into account in language processing. Paper [3] proposed a new representation which is the occurrence probabilities of n discontinued words in the same order because in natural language, 2 phrases with different amount of descriptive words in between may refer to the same thing. Authors proposed a novel technique with a Markov chain approach, it however yet share a common weakness with other language models, the terms occurrence independency assumption, which is further discussed in the next paper I reviewed. Paper [4] emphasized the importance of taking account of order dependency in syntax-based analysis. Authors proposed 2 modifications to the two models in Word2Vec skip gram and CBOW to include word order sensitivity in the algorithm. This is particular useful for part of speech tagging and dependency parsing.

To conclude my experiments, cosine similarities, Mahattan distance and Eculidean distance are good features in predicting news stances. Linear regression and Xgboost classification have fair performance in this task, yet results can be further improved by balancing the input data to allow enough data for the models to learn.

References:

[1] Adriaan M. J. Schakel, Benjamin J. Wilson. Measuring Word Significance using Distributed Representations of Words. Available at: https://arxiv.org/pdf/1508.02297v1.pdf (Accessed on: 3 April 2018)

[2] Charu C. Aggarwal1, Alexander Hinneburg2, and Daniel A. Keim2. On the Surprising Behavior of Distance Metrics in High Dimensional Space. Available at: https://bib.dbvis.de/uploadedFiles/155.pdf (Accessed on: 7 April 2018)

[3] Antoine Doucet, Helena Ahonen-Myka. A method to calculate probability and expected document frequency of discontinued word sequences. Available at: https://pdfs.semanticscholar.org/4f07/6339b4b80d33688066fec9c3052206244d91.pdf (Accessed on 8 April 2018)

[4] Wang Ling, Chris Dyer, Alan Black, Isabel Trancoso. Two/Too Simple Adaptations of Word2Vec for Syntax Problems. Available at: http://www.aclweb.org/anthology/N15-1142 (Accessed on 9 April 2018)