CMPE462: Machine Learning

Sentiment Analysis on IMDB User Reviews

Step 3 Report

Group Members:

- Halil Umut Özdemir - 2016400168
- Halil İbrahim Orhan - 2016400054
- Sertay Akpınar - 2016400075

Introduction

For the first part of this project, the aim is to collect data to obtain a system that makes sentiment analysis on IMDB user reviews. The second step of the project was the generation of the initial version of the sentiment analysis system. Finally, the third step is to enhance the results we obtained in step 2 via pre-trained embeddings. Our report includes 5 parts which are Data Preprocessing, Feature Extraction, Methodology, Results, Conclusion.

Step 3: Second Run

Data Preprocessing

As in step 2 we have preprocessed our data with several stages. Initially, contractions such as “He’s” or “We’re” are replaced with their long forms such as “He is” or “We are” using a precalculated contractions dictionary which is provided by an external module. Then, for all titles and bodies, all letters in text data are converted to lower-case. Next stage was the removal of information-free special patterns such as emails, URLs, punctuations, stopwords, and unnecessary blanks. Furthermore, non-alphabetic such as emojis, etc. Moreover, words that contain three repetitions of a single character are removed since they might be caused by typos.

Next stages are lemmatization and stemming the words. We used “nltk.stem.WordNetLemmatizer” to find where to cut a word. On the other hand, we used “nltk.porter.PorterStemmer” for the stemming process. We applied these 2 processes to our text data because suffixes of a word are often used in a close sense. Different from step 2, we also removed the words that are used 5 times or less in all texts. We applied this step after we saw the difference between validation results and test results. Because we thought that this difference is caused by the noise in the data, therefore we tried to decrease the noise by ignoring infrequent words. It also increased the accuracy in validation.

Feature Extraction

In the previous step, we used the Bag-of-Words model to represent syntactic information of user reviews. Also, NBSVM uses Naive Bayes posterior probabilities to represent each word in the document. However, in the previous step we do not remove the infrequent word from the document. In this step, we developed the Bag-of-Words method that we used in step 2 by removing the words in title that exist in less than 5 documents, and words in body that exist in less than 10 documents, as these words can cause noise problems. After that, we applied the ANOVA-F to select the most informative features from the calculated vectors. We give these features to the NBSVM model, as we did in step 2. By this operation the validation accuracy of the NBSVM model is increased from 0.6787 to 0.6960. We thought that the syntactic patterns can be learned by NB-SVM. Thus, in our final model we decide to use NBSVM as a feature extractor model. SVM learns a decision function with given data points. NBSVM uses the One-vs-All approach to handle multi-class classification problems. Consequently, we can generate 3 features by using the decision functions of NB-SVM. Also, to make predictions, NBSVM takes the maximum of these values. So, we also used pairwise differences of these 3 values as features. As a result, we encoded all the syntactic information in the user reviews on 6 features.

In addition to the syntactic parts of the user reviews, we used pre-trained sentence embeddings to obtain the semantic information. Sentence embeddings are calculated separately for both body and title parts to extract new features. Because most of the titles are formed by only one sentence, for title embeddings we use the whole title of a review as a sentence. In the case of body parts, because they are formed by more than one sentence and the order of sentences does not contain useful information. We want to obtain a single embedding vector for a user review body. Therefore, we tried various combinations of these sentence embeddings to choose optimum body embedding. These are,

1. Calculating sentence embedding of each sentence and taking their average,

2. Splitting the body parts into 200 word samples which has 50 word intersection and taking their average,
3. Taking the whole body part as a sentence and calculating its sentence embedding.

After comparing the accuracy results, we prefer to choose the first option as the optimum one. In title embeddings, again ANOVA-F is applied for feature selection. We used the features obtained from ANOVA-F in PCA to improve the quality of the features, as a result 3 features were obtained in PCA. We applied LDA to the same title embeddings and obtained 2 features. In the case of body embeddings, we tried several approaches. We tried ANOVA-F feature selection, PCA, and LDA as dimensionality reduction. After several trials, we obtained the best performance by using LDA and pure feature selection.

In addition to feature selection and dimensionality reductions, we used K-Means clustering to obtain new features from body and title embeddings. By using title and body embeddings separately, we trained 3 different K-Means for positive, negative and neutral reviews. After some calculations, we obtained the optimum number of clusters as 3. Then, we calculated the cosine similarity values between the center of the clusters, and the given data points. We obtained the new features by subtracting each cosine similarity vector from the other cosine similarity vectors. Then we tried to use these features on their own and with the help of PCA and LDA dimensionality reduction. As a result we obtained the best performance by reducing the dimensionality of these features with LDA.

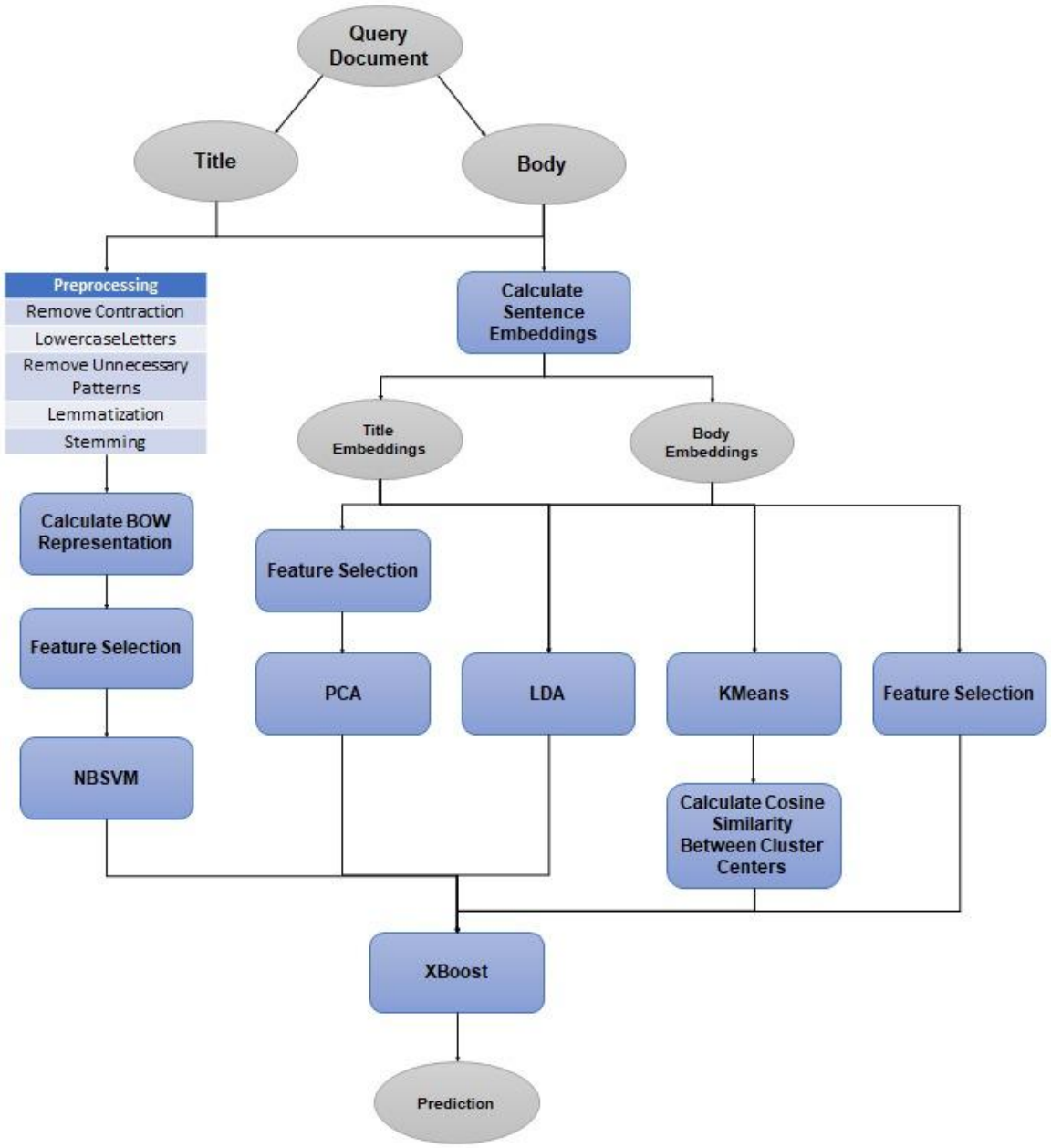
Up to this point, we mentioned the optimum features that we obtained. However, we tried various features to obtain this optimum set of features.We use singular value decomposition to obtain new features. We took the singular value decomposition of embedding vectors of each class separately, then chose the vectors related with the highest 5 singular values for each class. We assume that these 5 vectors for each class corresponds to the mostly used patterns in that class. Therefore, we calculated the cosine similarity between these 15 vectors and a data point. Additionally, we calculated the pairwise differences for the cosine similarities calculated for each vector. Then we tried to use calculated values as additional features. Even if it increased the accuracy, while we used this feature with the other features that we generated the resulting validation accuracy decreased. Therefore, we do not use these features in our final model.

We tried the dimensionality reduction algorithm in [1]. The dimensionality reduction algorithm utilizes a post processing algorithm(PPA). In the post processing, the mean of the word embedding matrix is subtracted from the word embedding matrix, PCA components are calculated and the top D components are eliminated. Then the main algorithm has the following steps. First, PPA is applied to the word embedding matrix. Then the resulting matrix is transformed using PCA. Finally, PPA is applied

again to the transformed matrix. Although we implemented the algorithm, there was not a notable advancement in the results.

Lastly, we tried to use some dictionaries that are generated for sentiment analysis. In these dictionaries there are points for each word given to determine its sentiment. By using these points we tried to calculate the average point of each review and use ity as a feature. However, these features also do not generate an increase in the accuracy score.

Methodology



As a result of this project the overall design of our sentiment analysis system is

shown in Figure X. In the step 2 of this project, we used Naive Bayes-Support Vector Machine (NBSVM) [2-3] to classify the IMDB user reviews. NBSVM is a combination of Naive Bayes and Support Vector Machine algorithms. It uses SVM for classification and it encodes the words in the document by using their class-based posterior probabilities. As can be seen in the results of the previous step, the resulting test accuracy is much less than the validation accuracy. We thought that the reason for this is the noise in our dataset. Therefore, we firstly removed the words that exist in less than 5 documents. As a result of this operation, the performance of NBSVM has developed. The validation accuracy score increased from 0.6787 to 0.6960.

After the good performance that we obtained with NBSVM, we decided to use it as a feature extractor for our new model. SVM learns a decision function to classify given data points. Additionally, the NBSVM model uses the One-vs-All approach for multiclass classification problems. Therefore, by using the syntactic information inside the user reviews, we obtained 3 features which represent the result of the decision function of 3 SVMs. As can be seen the final model that we used is XGBoost. Because XGBoost is a tree-based approach, to extract the maximum of these 3 values, we also give the pairwise differences, which also generates 3 features. The resulting 6 features are the features that represent the syntactic information of a given user review.

In the second part of this model, we used sentence embeddings to extract new features. To calculate sentence embeddings, we used a pre-trained model [4]. The sentence embeddings of the title and body are calculated separately. To calculate sentence embeddings of the title, all titles are embedded as a whole. In the case of body parts, we calculate the embedding of each sentence individually, and we take the average of all sentences to find the embedding of a user review body. After the calculation of these sentence embeddings , we used different techniques to obtain a set of features that gives the best performance in terms of accuracy. All the techniques that we tried are told in the *Feature Extraction* part of the report. Until now, only the ones that are used will be examined.

To obtain features from pre-trained embeddings, we used PCA, LDA, K-Means and ANOVA-F Feature Selection. We firstly used title embeddings to generate new features. Initially, we used PCA to obtain new features from title embeddings. As can be seen from the figure above, we firstly applied ANOVA-F feature selection to title embeddings and then by using PCA algorithm we decreased its dimension to 3. Secondly, we generated 2 features by using LDA on title embeddings. Lastly, we used an approach with K-Means, because the same procedure is used on body embeddings, this procedure will be explained later. In the case of body embeddings 2 features are generated by LDA operation. Additionally, by using ANOVA-F feature selection, 39 features are added to the features that will be used. Lastly, we used a procedure based on KMeans to obtain new features. This procedure is applied on

title and body embeddings separately. We initially separate the positive, negative and neutral embeddings and train a 3 KMeans clustering algorithm which generates 3 clusters for each class. After clustering, 9 cluster centers are obtained and we thought that these cluster centers represent the overall characteristic of their cluster. Therefore, cosine similarity between these 9 cluster centers are calculated for each data point. Then the pairwise differences of these cosine similarity measures are calculated. Lastly, to decrease its dimensionality, LDA is applied to the generated set of features. This operation is applied on title and body embeddings separately. Thus 4 features are obtained from this operation. As a result of the feature extraction part of the model, we obtained 56 features for the XGBoost Classifier.

After the feature extraction part, we tried different classifiers to obtain the best performance for our model. We tried SVM, XGBoost, LightGBM, and decision tree classifiers. As can be seen from the *Results* section, we obtained the best performance by using XGBoost. To prevent the model from overfitting, we used lots of shallow trees in the XGBoost model that we trained. So the maximum depth of each tree in the trained model is 2. Additionally, to utilize all features that we obtained with these sets of shallow trees, we randomly select 3 of these features to train each tree in the ensemble. All the hyperparameters of the classifier are fine-tuned empirically.

Results

By using the optimum set of features that we obtained, we try to use several classifiers to obtain best performance. In the following table, performances of these models are compared in terms of several performance metrics.

Model	Data	Class	Precision	Recall	F-1 Score	Accuracy
Step 2 NBSVM	Train	N	0.89	0.96	0.92	0.924
		Z	0.94	0.86	0.90	
		P	0.94	0.95	0.95	
	Val	N	0.70	0.74	0.72	0.679
		Z	0.60	0.48	0.53	
		P	0.72	0.82	0.77	
Step 3 NBSVM	Train	N	0.79	0.90	0.84	0.826
		Z	0.83	0.84	0.76	
		P	0.86	0.89	0.87	

	Val	N	0.69	0.76	0.72	0.696
		Z	0.63	0.50	0.56	
		P	0.75	0.83	0.79	
LightGBM	Train	N	0.71	0.89	0.79	0.782
		Z	0.82	0.64	0.72	
		P	0.84	0.82	0.83	
	Val	N	0.59	0.72	0.65	0.633
		Z	0.55	0.43	0.48	
		P	0.75	0.75	0.75	
SVM	Train	N	0.89	0.91	0.90	0.894
		Z	0.86	0.86	0.86	
		P	0.93	0.91	0.92	
	Val	N	0.69	0.67	0.68	0.692
		Z	0.59	0.57	0.58	
		P	0.79	0.84	0.81	

Model	**Data**	**Class**	**Precision**	**Recall**	**F-1 Score**	**Accuracy**
Random Forest	Train	N	0.83	0.88	0.85	0.844
		Z	0.81	0.77	0.79	
		P	0.89	0.89	0.89	
	Val	N	0.71	0.70	0.70	0.700
		Z	0.61	0.57	0.59	
		P	0.78	0.83	0.80	
XGBoost	Train	N	0.82	0.87	0.84	**0.830**
		Z	0.80	0.73	0.76	
		P	0.87	0.89	0.88	

	Val	N	0.75	0.73	0.74	**0.737**
		Z	0.65	0.61	0.63	
		P	0.80	0.87	0.83	

Conclusion

As a result of this project, we obtained a sentiment analysis system that classifies IMDB user reviews into 3 groups, Positive, Negative and Neutral. In the previous step, we implemented a classifier which is based on Naive Bayes-Support Vector Machine [2-3]. This version uses the syntactic information which is represented by Bag-of-Words representation. In this step, we improved our previous model by using different approaches. Initially, we removed the noise of the data by ignoring infrequent words in the user reviews. By doing this operation, the validation accuracy of our model increased. Then we combine the results obtained by NBSVM, and the features that we obtained with PCA, LDA, and KMeans from the pretrained embeddings of title and body parts of the user reviews. As a result, we generated a new set of features. Lastly, we trained an XGBoost Classifier with this feature set. This addition also increased the performance of our estimator. The resulting performance of our model can be found in the *Results* section.

Lastly, because of the size restrictions of Moodle, our pretrained model is uploaded to the following link:

https://drive.google.com/drive/folders/1_UBLNO7CMLqFciNh7IQ5HYshewMAZbgi?usp=sharing

Group Work

TASK	ASSIGNEE
Reporting of the Feature Extraction step, applying the LightGBM, applying the K-Means clustering	Sertay
Reporting Introduction and Data Preprocessing, Applying LDA to features, Implementation of SVM and Random Forest Model, Implementation of [1]	İbrahim
Implementation of Feature Engineering Steps, Initial Improvements on NBSVM, Fine Tuning Final Model XGBoost, Implementation of Main Scripts	Umut

References

[1] Raunak, V., Gupta, V., & Metze, F. (2019). Effective Dimensionality Reduction for Word Embeddings. *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*. https://doi.org/10.18653/v1/w19-4328

[2] https://github.com/prakhar-agarwal/Naive-Bayes-SVM/blob/master/nbsvm.py

[3] Wang, S., & Manning, C. (2012). *Baselines and Bigrams: Simple, Good Sentiment and Topic Classification*, Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics. https://www.aclweb.org/anthology/P12-2018.pdf

[4] Reimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. EMNLP/IJCNLP.