Big Data Report - IMDb reviews sentiment analysis
IDS 564 - Big Data

Project goal:

Apply “bag of words” and “Word2vec” on sentiment analysis of IMDb movie reviews to predict as a negative or a positive review.

Background:

Sentiment analysis is a challenging subject in machine learning. People express their emotions in languages in different ways and this can be hidden by sarcasm, ambiguity, and a play on words. Therefore, this can be very misleading to both human and computers. Given the data set of over 50k text data, we are going to conduct sentiment analysis on movie reviews from IMDb, which is a online database of information for films and the data we rely on has 100,000 multi-paragraph movie reviews, both positive and negative.

Team:

Ting Lan: Ting works for back-end by using Python to build up the model for machine learning in the project and did the data preprocessing.

Jonathan Nichols: Jonathan made the front-end of project which includes presentation notes, writing insight of data etc.

Both teammates perform well for this project and put much effort to finalize our discussion and conclusion.

Data sets:

labeledTrainData.tsv : 25,000 reviews with labels of “1” or “0” (positive /negative reviews)

testData.tsv: 25,000 reviews need to be predicted

unlabeledTrainData.tsv : 50,000 reviews without labels

- ❑ ***Tools (techniques, programming language,etc):***

Programming language: Python (Sklearn library for machine learning including parameter tuning)

- Data processing: numpy,pandas,nltk,tokenizer,BeautifulSoup
- Feature extraction and selection: CountVectorizer, tfidfVectorizer,pipeline
- parameter tuning: GridSearchCV
- Visualization : matplotlib
- Machine learning models: word2vec,MultinomialNB,GradientBoostingClassifier

- ❑ ***Further Details for tools used***

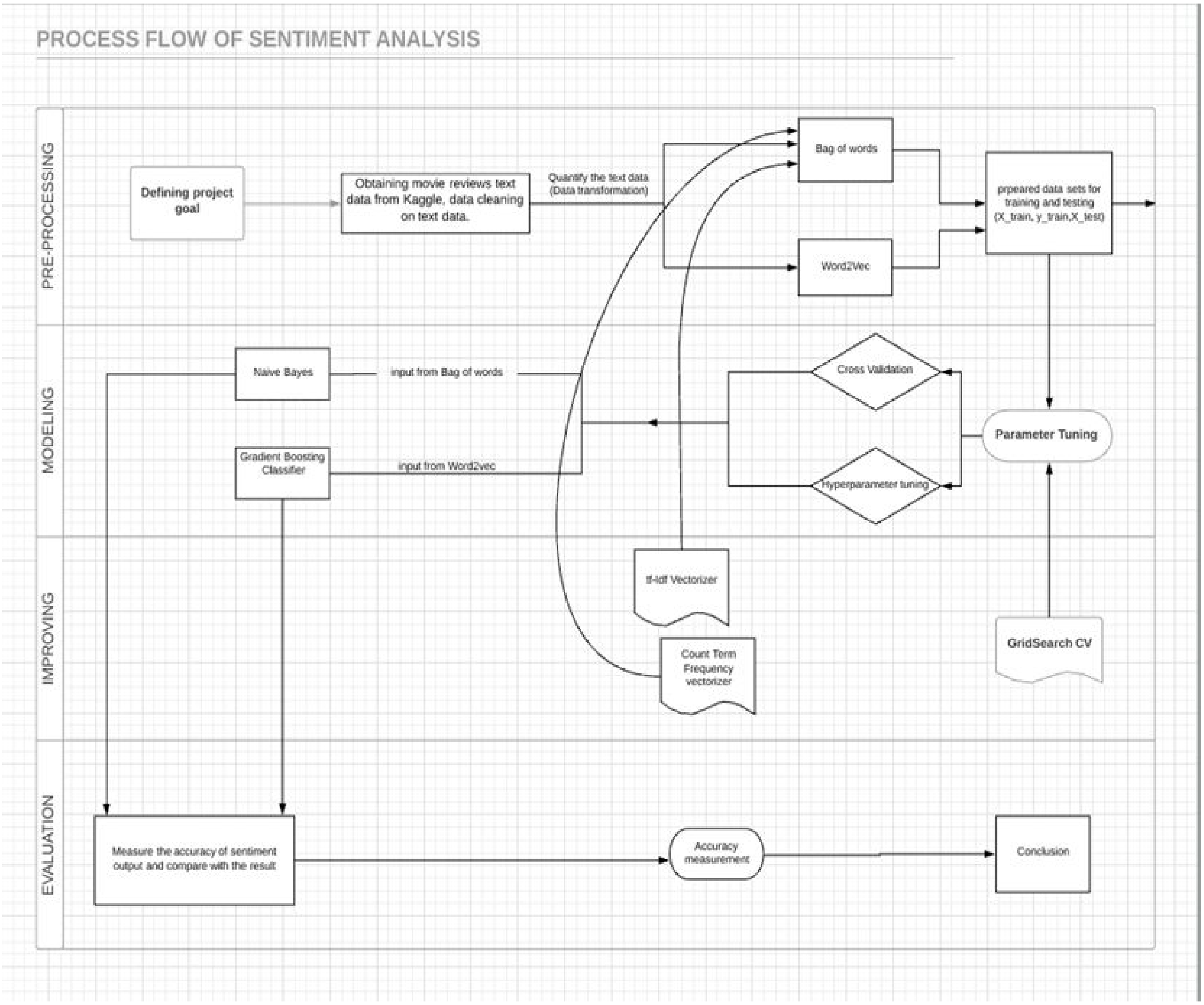
Challenges of modeling:

- Data cleaning on the reviews
- Tuning parameter-Parameters used in the model to maximize the performance of the model

Solutions:

- BeautifulSoup package on python
- Hyperparameters -gridSearch

Workflow and modeling process


PROCESS FLOW OF SENTIMENT ANALYSIS
PRE-PROCESSING
MODELING
IMPROVING
EVALUATION
Defining project goal
Obtaining movie reviews text data from Kaggle, data cleaning on text data.
Quantify the text data (Data transformation)
Bag of words
Word2Vec
prpeared data sets for training and testing (X_train, y_train,X_test)
Naive Bayes
input from Bag of words
Gradient Boosting Classifier
input from Word2vec
Cross Validation
Hyperparameter tuning
Parameter Tuning
tf-idf Vectorizer
Count Term Frequency vectorizer
GridSearch CV
Measure the accuracy of sentiment output and compare with the result
Accuracy measurement
Conclusion

According to the flow chart we created for our project, the simple process is as follows: defining the problem, data preprocessing, build up machine learning model for prediction, and evaluate the performance for each model by measuring the accuracy of output.

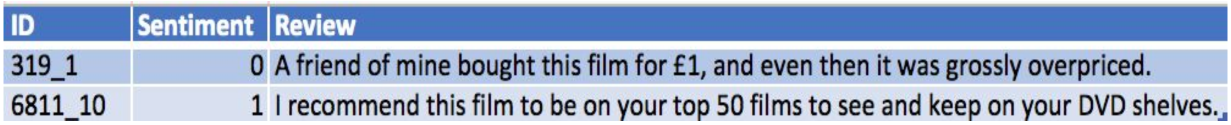
Modeling :

The project can be separated into 2 parts, sentiment analysis by using Bag of words and Word2Vec. In “Bag of words”, each term of text is quantify into vectors by using term frequency (CountVectorizer) and Tf-idf Vectorizer is also used to compare the model performance since this method considers the overall influence of each term in whole data set.

- **Bag of words:**

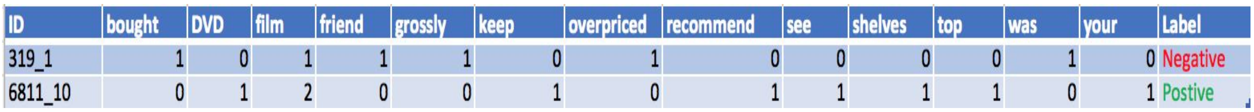
By using two rows of data in the training data set, the count vector is constructed in the following**:**

Original data:



ID	Sentiment	Review
319_1	0	A friend of mine bought this film for £1, and even then it was grossly overpriced.
6811_10	1	I recommend this film to be on your top 50 films to see and keep on your DVD shelves.

After using count vectorizer in python:

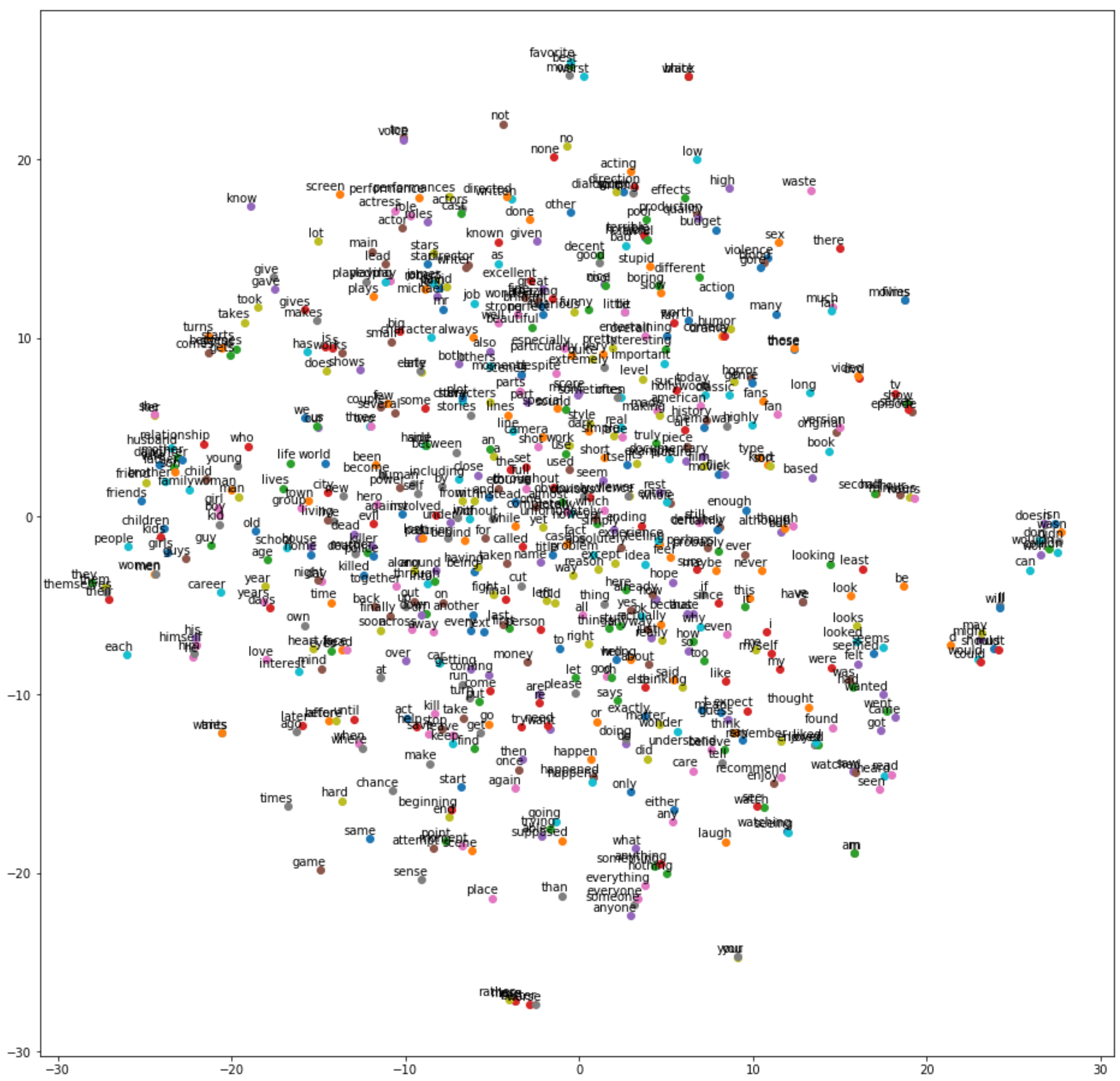


ID	bought	DVD	film	friend	grossly	keep	overpriced	recommend	see	shelves	top	was	your	Label
319_1	1	0	1	1	1	0	1	0	0	0	0	1	0	Negative
6811_10	0	1	2	0	0	1	0	1	1	1	1	0	1	Postive

The text vectors produced by this method is used as the input of the Naive bayes model, which predicts the output as 1 or 0, meaning positive or negative reviews in the test data.

- **Word2Vec**

By using Word2vec, the input data is split into tokens, and the output features vectors, which is the vocabulary corpus we built. All together, 300 features are created and each feature is mentioned more than 20 times in the movie reviews. In addition, by using numeric form, we can measure the similarity which is different from the bag of words technique. In the chart below, it shows the vocabulary corpus we built from the movie reviews, the more similar the meaning of two words, the closer the points are on the plot by using t-SNE. (a method to reduce dimension of features and project them on a lower dimension chart). The text vectors produced under the word2vec model would be used to calculate feature vectors for each word in the corpus we created. In the end, for each movie review, average feature vectors are computed as the input of classification model: Gradient Boosting Classifier.



Performance Improvement:

For each part of the project, we used grid search technique for parameter tuning to produce the best parameters for both predictive models. Grid search is a cross-validated method to select the best family of models over a grid of parameters.

Conclusion:

Ultimately, through using parameter tuning and model evaluation in the project we get accuracy results for both models. The Naive Bayes predictive model with “bag of words” gives us an accuracy of 86.3% and 86.5% for countvectorizer and tf-idf vectorizer respectively while the word2vec model with gradient boosting classifier reaches a similar accuracy of 85.5%. Sentiment analysis has a wide use of application and in this case, sentiment analysis on movie reviews can be used for better understanding of movies in the market and what are the driving factors of a good movie (based on market sentiment) can be learned for further research.

Bibliography

1. *Competition page on Kaggle: Bag of words meets Bag of popcorn.* https://www.kaggle.com/c/word2vec-nlp-tutorial#description
2. *Visualize word vector with t-SNE : Jeff Delany , 2017* https://www.kaggle.com/jeffd23/visualizing-word-vectors-with-t-sne