Online Review Spam Detection - Midway report

Group-2 : Tirth Patel, Harsh Kachhadia, Piyush Biraje
Department of Computer Science, North Carolina State University, Raleigh, NC
tdpatel2@ncsu.edu, hmkachha@ncsu.edu, pbiraje@ncsu.edu

1 Background and Introduction

Today, majority of the companies have become consumer focused. Customer experience is directly related to company's success. Customers are directed towards those products where they have good experience. Take the example of Amazon. People prefer to shop via Amazon because Amazon prioritizes customer experience. They do so by providing other customer reviews on products. This help people make good decisions. But lately we have seen an increase in fake reviews on many websites like Amazon, TripAdvisor, Yelp, etc.

It is this problem that many companies have tried to solve in order to give their customers correct information about the product. But its not easy to say which review is fake and which isn't. Machine Learning has helped to detect fake reviews with a certain degree of certainty.

In this project, we plan to come up with a model which will be able to predict if a given review is fake or not.

2 Method

For the first part of this project, our team plan to use a hotel review dataset obtained from Kaggle and create different Machine Learning models and then come up with a best model. We plan to create models based on Multinomial Naïve Bayes, Logistic Regression, SVM, SGR, Random Forest, etc.

For the second part of the project, we plan to create a web-scrapper and make our dataset of online reviews. After creating the dataset, we plan to apply the model selected from part one of the project on this dataset.

3 Experimetal Setup

In order to fulfill the first phase of our project which is, selecting the best supervised machine learning algorithm for review/opinion classification into deceptive or truthful, we utilized the following data, algorithms, and model evaluation methods:

3.1 Data:

`https://www.kaggle.com/rtatman/deceptive-opinion-spam-corpus?select=deceptive-opinion.csv`

Data Exploration: This dataset contains hotel reviews from 20 chicago hotels.

Our dataset has 1600 records - 800 labelled as truthful and 800 as deceptive reviews.
Data that we will be using - "Text" and "deceptive" columns will be used for our project.

Dataset has the following columns:
1. Deceptive - class label - “deceptive” or “truthful”
2. Hotel - describes hotel for which review is given
3. Polarity - positive or negative
4. Source - the website on which the hotel review was posted
5. Text - the review text

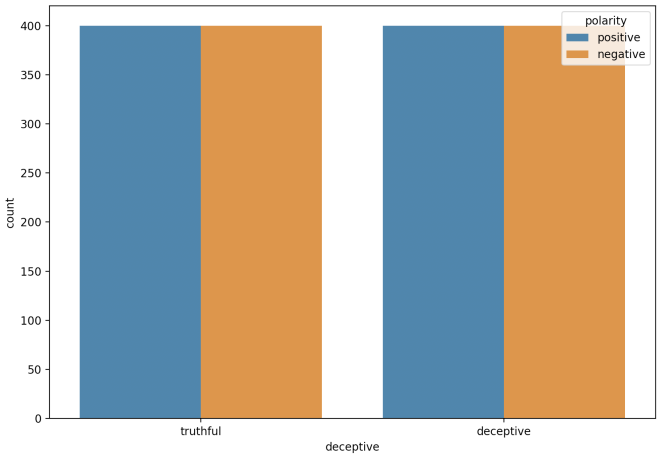

Figure 1: Accuracy Table

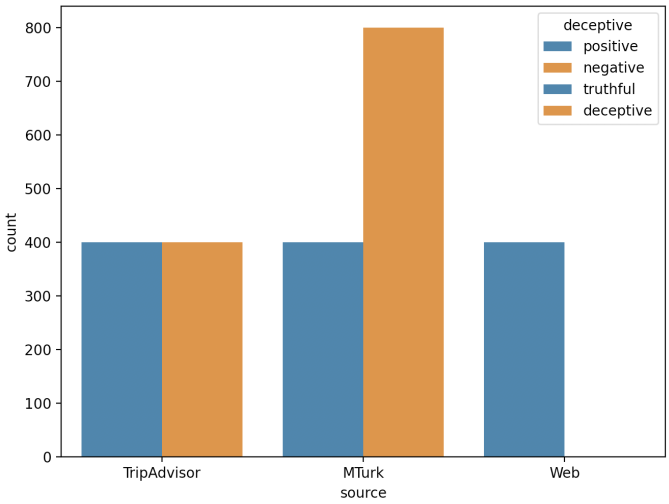

Figure 2: Accuracy Table

Input Data Preprocessing: The “Text” column has been preprocessed with NLTK Library and Regular expressions in python. Following text-preprocessing has been done.

1. Convert text to lowercase
2. Remove - punctuation, text in square brackets, non-alphanumeric characters, hyperlinks, special characters, new line characters, digits, extra space between words, front or trailing spaces in text
3. Removal of stopwords from the text.

Output Label Preprocessing: Before proceeding with implementing our algorithms, we have encoded our output class labels (deceptive or truthful) to numeric values with the help of sklearn Label Encoder.

Model Evaluation Methods implemented:

1. Accuracy
2. K-Fold CV - 5-CV, 8-CV, 10-CV
3. Holdout Method (train,test, split) (0.7-train, 0.3 test)

3.2 Algorithms Implemented for experimentation:

1. Multinomial Naive Bayes Classifier
After performing data preprocessing on the "Text" column of the data, which contains reviews about hotels, we implemented a pipeline which performs 2 tasks: 1) Convert the pre-processed text in the "Text" column to TF-IDF feature vectors. 2) Train the sklearn provided Multinomial Naive Bayes Classifier(default hyperparameters) on the TF-IDF feature vectors. This pipeline was also subsequently used for prediction on the test data.

This pipeline was utilized to try out different models of Multinomial Naive Bayes Classifier trained on different datasets obtained by performing the following model evaluation techniques on our preprocessed data.

K-Fold Cross Validation (K=5,8,10), Holdout Method (TTS) (0.7-train, 0.3 test).....(Eq. 1)

The accuracy score of each of these models has been recorded in the results table in the results section.

2. Stochastic Gradient Descent
After experimenting with different models of Multinomial Naive Bayes Classifier, we implemented another pipeline which performs 2 tasks: 1) Convert the pre-processed text in the "Text" column to TF-IDF feature vectors. 2) Train the sklearn provided SGD(stochastic gradient descent) Classifier with the following hyperparameters on the TF-IDF feature vectors.

```
SGDClassifier(loss='hinge', penalty='l2',alpha=1e-3, random_state=42, max_iter=5
,tol=None)
```

This pipeline was also subsequently used for prediction on the test data.

Similar to Multinomial Naive Bayes, this pipeline was also utilized to try out different models of SGD(stochastic gradient descent) Classifier trained on different datasets obtained by performing the model evaluation techniques in (Eq.1) on our preprocessed data. The accuracy score of each of these models has been recorded in the results table in the results section.

3. Logistic regression
After experimenting with different models of SGD(stochastic gradient descent) Classifier, we implemented another pipeline which performs 2 tasks: 1) Convert the pre-processed text in the "Text" column to TF-IDF feature vectors. 2) Train the sklearn provided Logistic Regression Classifier with the following hyperparameters on the TF-IDF feature vectors.

```
LogisticRegression(n_jobs=1, C=1e5)
```

This pipeline was also subsequently used for prediction on the test data.

Similar to Multinomial Naive Bayes, this pipeline was also utilized to try out different models of Logistic Regression Classifier trained on different datasets obtained by performing the model evaluation techniques in (Eq.1) on our preprocessed data. The accuracy score of each of these models has been recorded in the results table in the results section.

4. SVC (Linear SVM) Classifier
After experimenting with different models of Logistic Regression Classifier, we implemented another pipeline which performs 2 tasks: 1) Convert the pre-processed text in the "Text" column to TF-IDF feature vectors. 2) Train the sklearn provided SVC (Linear SVM) Classifier with the following hyperparameters on the TF-IDF feature vectors.

```
LinearSVC(loss='hinge', C=5, random_state=42)
```

This pipeline was also subsequently used for prediction on the test data.

Similar to Multinomial Naive Bayes, this pipeline was also utilized to try out different models of SVC (Linear SVM) Classifier trained on different datasets obtained by performing the model evaluation techniques in (Eq.1) on our preprocessed data. The accuracy score of each of these models has been recorded in the results table in the results section.

5. Random Forest Classifier
After experimenting with different models of SVC (Linear SVM) Classifier, we implemented another pipeline which performs 2 tasks: 1) Convert the pre-processed text in the "Text" column to TF-IDF feature vectors. 2) Train the sklearn provided Random Forest Classifier with default hyperparameters on the TF-IDF feature vectors. This pipeline was also subsequently used for prediction on the test data.

Similar to Multinomial Naive Bayes, this pipeline was also utilized to try out different models of SVC (Linear SVM) Classifier trained on different datasets obtained by performing the model evaluation techniques in (Eq.1) on our preprocessed data. The accuracy score of each of this models has been recorded in the results table in the results section.

4 Results

The results from the implementation of various models of algorithms in the experimental setup section have been tabulated here:

	Model	TTS	CV-5	CV-8	CV-10
0	Multinomial Naive Bayes	0.854167	0.85875	0.864375	0.860625
1	SGD	0.885417	0.87625	0.883750	0.882500
2	Logistic Regression	0.885417	0.87250	0.874375	0.876875
3	SVC (Linear SVM)	0.877083	0.86750	0.874375	0.872500
4	Random Forest	0.829167	0.85125	0.841875	0.855625

Figure 3: Accuracy Table

5 Conclusion

Out of the 5 algorithms implemented so far, the stochastic gradient descent (SGD) algorithm has returned the best results with consistently highest accuracy(88.5%) across all model evaluation techniques. In future we plan to proceed with SGD Classifier to tag our collected data (collected using web scraper.)

6 References:

1. https://www.researchgate.net/publication/325075174_Detection_of_fake_online_hotel_reviews
2. https://arxiv.org/pdf/1903.12452.pdf
3. https://www.cs.uic.edu/~liub/FBS/opinion-spam-WSDM-08.pdf