Fake News Classification

Online Social Network Analysis Final Project

Chandni Patel
A20455322

Omkar Pawar
A20448802

INTRODUCTION

Social media has become one of the major resources for people to obtain news and information. For example, it is found that social media now outperforms television as the major news source. However, because it is cheap to provide news online and much faster and easier to disseminate through social media, large volumes of fake news or misinformation are produced online for a variety of purposes, such as financial and political gain. The extensive spread of fake news/misinformation can have a serious negative impact on individuals and society: (i) breaking the authenticity balance of the news ecosystem; (ii) intentionally persuading consumers to accept biased or false beliefs; and (iii) changing the way people interpret and respond to real news and information. Therefore, it is important to detect fake news and misinformation in social media.

PROBLEM STATEMENT

We formally define the task as follows. Given the title of a fake news article A and the title of a coming news article B, participants are asked to classify B into one of the three categories:

- **agreed**: B talks about the same fake news as A.
- **disagreed**: B refutes the fake news in A.
- **unrelated**: B is unrelated to A.

DATA

We are provided with 3 CSV files:

- **train.csv**: Training data
- **test.csv**: Test data
- **sample submission.csv**: Expected submission format

The training data includes the “label” of each news pair, while the test data doesn’t. Validation data can be split from train.csv. Training data should be used to train a classifier and evaluate a model's performance with the validation data. Finally, by using the trained model, we are required to predict the results for the test data. The format of our output file should be the same as “sample submission.csv” with our prediction replaced in the“label” column.

The columns in train and test data are as follows:

• **id**: the id of each news pair.

• **tid1**: the id of fake news title 1.

• **tid2**: the id of news title 2.

• **title1 en**: the fake news title 1 in English.

• **title2 en**: the news title 2 in English.

• **label**: indicates the relation between the news pair: agreed/disagreed/unrelated.

DATA PREPROCESSING

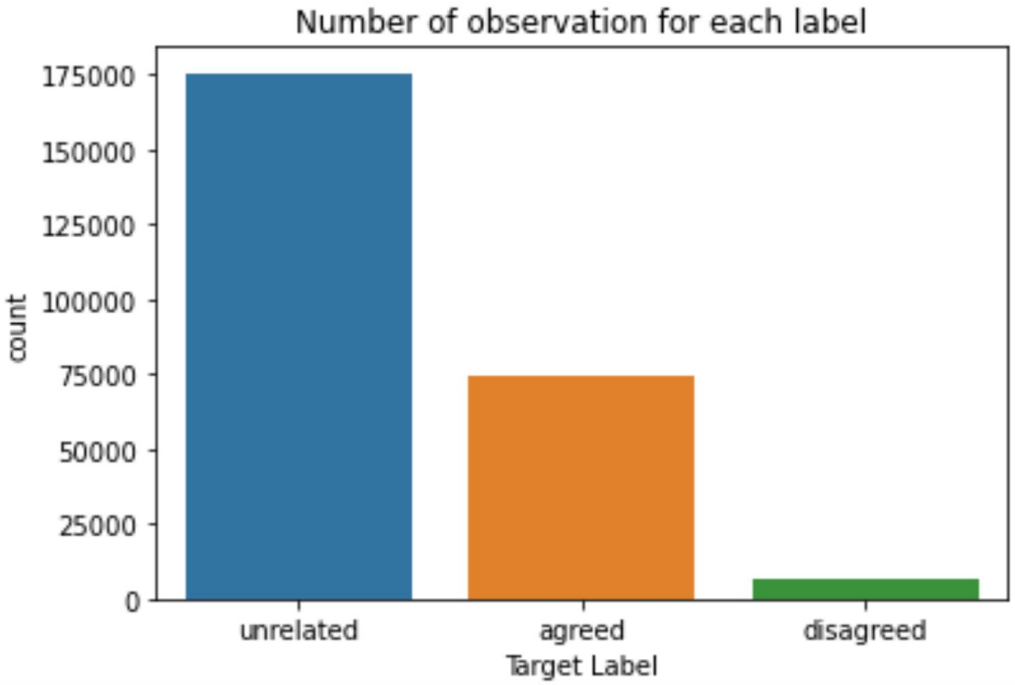
Basic preprocessing steps are required before using the data:

1. Convert to Lowercase
2. Remove punctuations
3. Remove single character if any
4. Remove stop words
    - Eliminate frequently used words which does not add much meaning to the

sentence, increase processing time and add memory in the data
    - Example: the, is, there, you, etc.
5. Convert numbers to words
    - Example: **110** is converted to "**one hundred and ten**"
6. Lemmatization to get root words
    - Base or dictionary form of a word, which is known as the lemma to remove inflectional endings.
    - Example: "**working**" is converted to "**work**"

APPROACH

Initially we did some **exploratory data analysis** to how the data is distributed. The end goal of the assignment is to classify the given pair of strings into three categories. The distribution of the categories that we have in the training data is as follows.



From the count plot it can be seen that the labels are distributed unevenly. We have a lot of observations labelled as unrelated. After having a brief understanding of data, we took different approaches to find the solution.

FEATURE EXTRACTION

SIMILARITY BETWEEN TEXTS

One of the approaches was to check the similarity between texts and use these values to classify them into specific classes based on the similarity value. We used cosine similarity to create the feature. This was a very simplistic approach where we just classified data into these classes.

We just compared the two headlines of the news from given data traversing each row. We have to take a step further and try to build models which can sense the meaning of sentences and not just compare the words occurring in it.

BAG OF WORDS

A bag-of-words model is a way of extracting features from text for use in modeling, such as with machine learning algorithms. Using this approach we tried to create features for each column “title 1” and “title 2”. We can train our model on these features and predict the outcome of testing data.

A bag-of-words is a representation of text that describes the occurrence of words within a document. It involves two things: (i) A vocabulary of known words; (ii) A measure of the presence of known words.

We created a Naive Bayes model using a bag of words approach and got an accuracy of around 73%. It shows that the model is poorly performing in predicting the labels of observations. We can understand why the model performs badly and there are a lot of things we can improve.

When we create a bag of words, we ignore the order of words in a sentence. Discarding word order ignores the context, and in turn meaning of words in the document (semantics). Context and meaning can offer a lot to the model, that if modeled could tell the difference between the same words differently arranged. Hence it is important that we create features which keep the context of each word in mind too. This leads us to the further step.

TF-IDF

In TF-IDF, the term frequency measures the frequency of a word in a document and then divided by the total number of words in the document for normalization. When we vectorize the document, we do it on the vocab list of all possible words in the corpus so that we don't just consider the words present in a single document. Document frequency is the number of documents in which a particular word is present which is also normalized by dividing it with the total number of documents. Inverse of document frequency measures the informativeness of a particular term across documents. Common recurring words like stop words will have low value and the value increases for important words. Final TF-IDF is obtained by multiplying TF and IDF.

We extract features for modeling using TF-IDF for each column "title 1" and "title 2" and then add them to create final features. We can train our model on these features and predict the outcome of testing data.

DATA MODELING

Once we have done the feature extraction, we plan to implement a few machine learning classification models. The models that we decided to go with are as follows:

Naive Bayes

Naive Bayes classifiers are a collection of classification algorithms based on Bayes' Theorem. Bayes' Theorem finds the probability of an event occurring given the probability of another event that has already occurred. It is the most common model used in natural language processing. The vectors that we get from feature extraction have a large dimension. Naive Bayes trains faster on such data and hence that is one reason we used it. It can also be used as a benchmark to evaluate and compare other models to see how they perform. As discussed above, we modeled our data using Naive Bayes and bag of words. Due to the poor performance of this model, we have decided to try other models as well. One of the models is Logistic Regression.

Multinomial Logistic Regression

Multinomial logistic regression can be used for categorical targets. We can train a logistic regression model using the features created through TF-IDF and then make predictions on test data. In our case, we have more than 3 classes. Therefore, Softmax regression can be used as it is a version of logistic regression for multi-class classification. Basically, it normalizes input values into vectors of values with a probability distribution.

When we modeled the logistic regression using TF-IDF generated features, we got accuracy of 81% on the test data. This is a significant improvement from Naive Bayes and it is clear that the logistic regression fits the data well.

Multi Layer Perceptron Classifier

MLP is a type of artificial neural network (ANN). Simplest MLP consists of at least three layers of nodes: an input layer, a hidden layer and an output layer. This model optimizes the log-loss function using LBFGS or stochastic gradient descent.

While creating the network, we used following parameter values

- Activation = relu
- Maximum iterations = 20
- Number of hidden Layers = 100
- Solver = Adam
- Batch Size = 128

Using this set of parameters, the model took longer time to train but showed improvement in performance. The accuracy values on training and testing data are shown in the next section.

MODEL EVALUATION

For evaluating the performance of the models, we used an accuracy metric. The values for training and testing accuracy for each model are mentioned below. These values are calculated using TF-IDF vectors as a feature in each model.

MODEL NAME	TRAINING ACCURACY	TESTING ACCURACY
NAIVE BAYES	78.92 %	76.61 %
LOGISTIC REGRESSION	85.48 %	80.78%
MULTILAYER PERCEPTRON	99.82 %	85.15%

CONCLUSION

We implemented different approaches for data preprocessing and modeling.

- It was observed that features generated using TF-IDF method gave better results on every model than cosine similarity features. It can be justified as similarity calculations only take into account one feature but TF-IDF gives more information than that and creates better features.
- The Multilayer Perceptron Classifier gave better performance than two other models (Naive Bayes and Logistic Regression).
- In future, we plan to use both similarity and TF-IDF together as our features to create models.
- The data is highly imbalanced with respect to the labels. Due to this the model might be getting trained better on “unrelated” observation but struggles to make correct predictions of other labels. Hence, in order to get better results, we can use more data which is balanced and has more observations with “agreed” and “disagreed”.

REFERENCES

1. https://machinelearningmastery.com/gentle-introduction-bag-words-model/
2. https://machinelearningmastery.com/classification-as-conditional-probability-and-the-naive-bayes-algorithm/
3. https://towardsdatascience.com/tf-idf-for-document-ranking-from-scratch-in-python-on-real-world-dataset-796d339a4089
4. https://towardsdatascience.com/logistic-regression-detailed-overview-46c4da4303bc
5. https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html