Opinion-Fact Classification

Kikkuru Sarath Chandra Reddy(MT19037)
Kasarla Mani Kumar Reddy (MT19065)
Kastala Murali Krishna (MT19132)

Problem Statement

- In the present-day technology, huge amount of data is being generated every day. So, it's turning out to be a challenging task to handle text-based data.
- In the world of text-based sentences it is not that simple to differentiate between fact and opinions.
- So, our project is to build the model that classifies/identifies facts from/and opinions in the given text by using various machine learning and deep learning techniques.

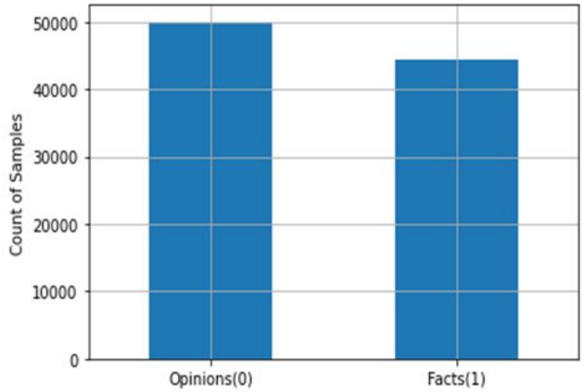
Dataset Detailing

- The dataset we considered was on movies domain.

- The **Facts** are from Kaggle site (plots) and reviews for the movies are considered as **opinions** which are collected from IMDB site.
  https://www.kaggle.com/rounakbanik/the-movies-dataset?select=movies_metadata.csv

- The dataset contains 94,379 samples which are facts or opinions.

- Dataset has reviews count of 50,000 whereas facts of 44,379.



Preprocessing steps used

- Stop-Word removal
- Case Conversion
- Tokenization
- Lemmatization
- Removal of alphanumeric words and special characters.
- Removal of words of length less than 3.

Word Embeddings Used

- **Bag of Words (BOW)**
  - In the BOW approach, each word in the sentence or text is replaced by the count of it occurences in the corresponding training sentence.
- **Term Frequency Inverse Document Frequency (TFIDF)**
  - Each word in the sentence is substituted by its tf-idf score, which reflects how important that word is to a document (each sample is a document here) in a collection or corpus.
- Both the approaches do not take any positional information into consideration.
- **Rank of the word in the vocabulary (used for LSTM)**
  - Here words in vocabulary are sorted in descending order of their occurrences and rank of the word is index of word in the list+1.

Learning techniques used

- K-NN (K nearest neighbours) with GridSearchCV for hyper-parameter tuning
- Naive Bayes with GridSearchCV for hyper-parameter tuning
- Decision Trees with GridSearchCV for hyper-parameter tuning
- SVM (Support vector machines) with GridSearchCV for hyper-parameter tuning
- LSTM (Long Short Term Memory)
  - Architecture

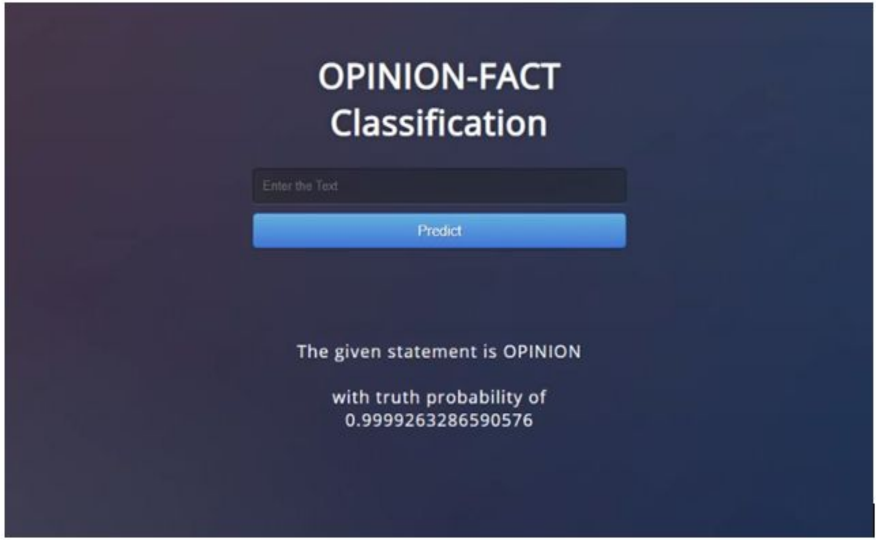
    1. Embedding Layer
    2. LSTM cell with 20 activation units
    3. Output layer with 2 activation units with softmax activation

Results (plots & analysis in report)

Model Implemented	Word-Embedding Used	Precision achieved on Test Data	Recall achieved on Test Data	F-Score achieved on Test Data	Accuracy achieved on test data
K-NN (baseline)	TF-IDF	0.6387	0.506	0.351	50.6%
K-NN	BOW	0.832	0.754	0.739	75.45%
Naïve Bayes	BOW	0.814	0.795	0.792	79.50%
Naïve Bayes	TF-IDF	0.811	0.788	0.7844	78.85%
Decision Trees	BOW	0.9062	0.9050	0.9046	90.46%
Decision Trees	TF-IDF	0.9192	0.9180	0.9176	91.76%
SVM	BOW	0.9576	0.956	0.9569	95.7%
SVM	TF-IDF	0.9542	0.953	0.9539	95.4%
LSTM	Rank of word in the vocabulary	0.9866	0.9870	0.9867	98.62%

Deployment

- The best performing model (LSTM) is deployed using flask-web framework on local host. The screenshot for the same can be seen below.(the text vanishes from text box after clicking on predict)



Conclusion

- We achieved best results with the movies dataset where facts are from Kaggle site (plot) and reviews of the movie is considered as opinion collected from IMDB site.
- The main challenge we faced in this problem is collection of data.
- We tried out different Machine Learning algorithms and also LSTM (deep learning algorithm) for this task and achieved good performances.
- Further the same methodology can also be used to detect bias (opinions) in news articles also (future work).