NLP-FAKE NEWS DETECTION

Fake news detection is the most pressing problem in the real world. Our main aim is to perform sentiment analysis on the Fake news detection dataset consisting of 10,270 statements and we are simplifying labels as binary labels 'Real' or 'Fake'. We use a Linear SVM classifier to predict labels and perform analysis on the model.

Simple data input and pre-processing

Load the data using CSV reader, consider only label and text columns and append it into raw data. Here the parse data line function splits the data into text and labels where labels are converted using the convert label function based on the degree of their fakeness into real or fake. Split the raw data into 80%train data and 20% test data. Trained data is preprocessed using a pre-process function. Here word tokenizer from the nltk library splits the text into tokens of words using a defined punctuation library and Porter Stemmer converts the tokenized text words into their root form and this is returned in the form of a list.

Simple feature extraction

These lists of tokens are assigned with some weights using the feature-vector function. Global feature dictionary helps us to track the number of features in the dataset and a unique index is assigned if a word appears only once if a word is present in the dictionary, then its index is used. Local feature dictionaries calculate the term frequency of the words i.e., the number of times a word occurs in the document by a total number of words in the dictionary. Term frequency is used to normalize the text in the document. Feature vector returns the dictionary consisting of words and their frequencies in the corpus.

Cross-validation on training data

Cross-validation is a widely used technique to identify how effectively our machine-learning model validates unseen data. In our model training data set is trained and tested using k-fold cross-validation where data is divided into 10 folds, for each fold training and validation data is created. In the first fold(i=0), validation data is tested using the index [0: 0+fold-size] and the remaining data is trained using a Linear SVM classifier model. Labels are predicted from the classified model in the defined function predict label using the test data. Precision, recall, f score, and support are collected from the predicted and defined labels of the validation data in the first fold. A similar process is continued for all the folds. Thereafter, the list of average precision, recall f-score, and accuracy for all the folds are returned.

ERROR ANALYSIS

Identified False positives and False negatives through confusion matrix.
Considering training and validation data for the first fold where "i=0" to evaluate false positives and false negatives.

After looking at the FP file, some observations:

- Claims involving numbers and proportions are common (e.g. "Half of illegal immigrants come on legal visas and then overstay.", "President Barack Obamas spending drove us $5 trillion deeper in debt.")- could add a number feature which picks up both written numbers and digits
- Some punctuation could be useful, like dollar signs for money claims, however also need to look at separation (e.g. "money.They") and also use of square brackets from journalist/corpus annotators may need removing/cleaning

After looking at the FN file, there are several observations:

- Claims about groups and/or foreign countries (to the USA) are prevalent (e.g. "Blocking travel from countries with Ebola should be possible because President Barack Obama has sealed off Israel in the past."
- Possibly sentiment is stronger than in the FNs with stronger claims, possibly exaggerated ("Under Greg Abbott, Texas four-year-olds would be forced to undergo standardized tests." "Poll after poll after poll shows me beating Hillary.")
- Many numerical claims again, though in different formats. May be more extreme numerical claims or vague compared to the other REAL TPs - in either way, given different formats need to join represent numerical claims in a similar way whether in figures or in words if there is something about the nature of the claims which means they're fake

Optimising pre-processing and feature extraction.

Extracting all the features from the data set in the “parse_data_line” function. For sentiment analysis, using Sentiment Intensity Analyzer to assign polarities for a statement positive, negative, neutral, and compound returning these values to the load function. In the “pre_process” function word tokenizer splits the text into tokens of words and snowball stemmer converts the tokenized text words into their root form and it is returned in the form of a list. Improvising the code by removing punctuation marks using the regular expression function.

For the feature extraction, I’m using TF-IDF (Term Frequency and inverse term frequency). In the Classifier, function Pipeline does feature extraction and uses a machine learning model for classifying labels. TF-IDF-tokens are returned from the "preprocess_all_features" function where stemming, lemmatization, lowercasing, and other preprocessing converts the tokens into unigram, bigrams, and trigrams and for assigning weights for the feature true I am opting TF-IDF method opting “smooth_idf” method for smoothing unknown tokens which appear only in test data.

Using other metadata in the file

From the tab separated dataline extracting all the additional features from the fake news data. It uses the "convert_labels" function to convert labels into fake or real. Sentiment intensity analyzer assigns a polarity for each statement. Parse_data_line returns labels, statement, subject, speaker, speaker_job and other features along with the polarities in the form of string for our convenience.

Result table for the test data

Features	Precision score	Recall score	Fscore
Unigrams	0.60	0.61	0.60
ngram(1,3) + additional features	0.64	0.64	0.63
ngram(1,3) + addition features + Sentiment Intensity Analyzer polarity scores.	0.70	0.70	0.70