NATURAL LANGUAGE PROCESSING

MOHAN M J

Introduction

- Natural Language Processing (or NLP) is applying Machine Learning models to text and language
- NLP is a set of techniques for approaching text problems
- Teaching machines to understand what is said in spoken and written word is the focus of NLP
- Whenever you dictate something into your smartphone that is then converted to text, an NLP algorithm is in action

NLP Uses

- To predict if the text review is a good one or a bad one
- To predict some categories of the articles you are trying to segment
- To predict the genre of the book
- To build a machine translator or a speech recognition system
- To classify language using classification algorithms
- Most of NLP algorithms are classification models, and they include Logistic Regression, Naive Bayes, CART etc.

Sentiment Analysis

- Sentiment analysis is a challenging subject in machine learning
- People express their emotions in language that is often obscured by sarcasm, ambiguity, and plays on words, all of which could be very misleading for both humans and computers

Bag of Words(BoW) Model

- A problem with modeling text is that it is messy, and techniques like machine learning algorithms prefer well defined fixed-length inputs and outputs.
- Machine learning algorithms cannot work with raw text directly; the text must be converted into numbers.
- In language processing, the vectors x are derived from textual data, in order to reflect various linguistic properties of the text
- The bag-of-words model is a way of representing text data when modeling text with machine learning algorithms.
- The bag-of-words model is simple to understand and implement and has seen great success in problems such as language modeling and document classification.

BoW - Intuition

- A bag-of-words is a representation of text that describes the occurrence of words within a document. It involves two things:
  - A vocabulary of known words.
  - A measure of the presence of known words.
- It is called a "*bag*" of words, because any information about the order or structure of words in the document is discarded.
- The model is only concerned with whether known words occur in the document, not where in the document.

BoW Model - Step 1: **Collect Data**

- This model used to preprocess the texts before fitting the classification algorithms on the observations containing the texts
- Below is a snippet of the first few lines of text from the book "A Tale of Two Cities" by Charles Dickens, taken from Project Gutenberg.
  - *It was the best of times,*
  - *it was the worst of times,*
  - *it was the age of wisdom,*
  - *it was the age of foolishness,*
- For this small example, let's treat each line as a separate "document" and the 4 lines as our entire corpus of documents.

Step 2: Design the Vocabulary

- Now we can make a list of all of the words in our model vocabulary.
- The unique words here (ignoring case and punctuation) are:
  - "it"
  - "was"
  - "the"
  - "best"
  - "of"
  - "times"
  - "worst"
  - "age"
  - "wisdom"
  - "foolishness"

It was the best of times,
it was the worst of times,
it was the age of wisdom,
it was the age of foolishness,

- That is a vocabulary of 10 words from a corpus containing 24 words.

Step 3: Create Document Vectors

- The next step is to score the words in each document.
- The objective is to turn each document of free text into a vector that we can use as input or output for a machine learning model.
- Because we know the vocabulary has 10 words, we can use a fixed-length document representation of 10, with one position in the vector to score each word.
- The simplest scoring method is to mark the presence of words as a boolean value, 0 for absent, 1 for present.
- Using the arbitrary ordering of words listed above in our vocabulary, we can step through the first document ("It was the best of times") and convert it into a binary vector.

Contd..

- Sentence - "*It was the best of times*"
- The scoring of the document would look as follows:
  - "it" = 1
  - "was" = 1
  - "the" = 1
  - "best" = 1
  - "of" = 1
  - "times" = 1
  - "worst" = 0
  - "age" = 0
  - "wisdom" = 0
  - "foolishness" = 0

"*It was the best of times*"

Contd..

- "it was the worst of times" = [1, 1, 1, 0, 1, 1, 1, 0, 0, 0]
- "it was the age of wisdom" = [1, 1, 1, 0, 1, 0, 0, 1, 1, 0]
- "it was the age of foolishness" = [1, 1, 1, 0, 1, 0, 0, 1, 0, 1]
- All ordering of the words is nominally discarded and we have a consistent way of extracting features from any document in our corpus, ready for use in modeling.
- New documents that overlap with the vocabulary of known words, but may contain words outside of the vocabulary, can still be encoded, where only the occurrence of known words are scored and unknown words are ignored.

Managing Vocabulary

- As the vocabulary size increases, so does the vector representation of documents.
- In the previous example, the length of the document vector is equal to the number of known words.
- Imagine that for a very large corpus, such as thousands of books, that the length of the vector might be thousands or millions of positions. Further, each document may contain very few of the known words in the vocabulary.
- This results in a vector with lots of zero scores, called a sparse vector or sparse representation.

Contd..

- Sparse vectors require more memory and computational resources when modeling and the vast number of positions or dimensions can make the modeling process very challenging for traditional algorithms.
- There are simple text cleaning techniques that can be used as a first step, such as:
  - *Ignoring case*
  - *Ignoring punctuation*
  - *Ignoring frequent words that don't contain much information, called stop words, like "a," "of," etc.*
  - *Fixing misspelled words.*
  - *Reducing words to their stem (e.g. "play" from "playing") using stemming algorithms.*

Dataset - **Movie Review**

- The Movie Review Data is a collection of movie reviews retrieved from the imdb.com
- The dataset is comprised of 1,000 positive and 1,000 negative movie reviews
- Reviews are stored one per file with a naming convention cv000 to cv999 for each neg and pos.

Data Preparation

- Separation of data into training and test sets.
- Loading and cleaning the data to remove punctuation and numbers.
- Defining a vocabulary of preferred words.

Split into Train and Test Sets

- We are developing a system that can predict the sentiment of a textual movie review as either positive or negative.
- We will use the last 100 positive reviews and the last 100 negative reviews as a test set (100 reviews) and the remaining 1,800 reviews as the training dataset.
- This is a 90% train, 10% split of the data.
- The split can be imposed easily by using the filenames of the reviews where reviews named 000 to 899 are for training data and reviews named 900 onwards are for testing the model.

Preprocessing data

Loading and cleaning review data

1. Split tokens on white space.
2. Remove all punctuation from words.
3. Remove all words that are not purely comprised of alphabetical characters.
4. Remove all words that are known stop words.
5. Remove all words that have a length <= 1 character.

THANK YOU