Bag-of-Words

Overview

1.The Problem with Text

2.What is a Bag-of-Words?

3.Example of the Bag-of-words Model

4.Advantages and Disadvantages

The Problem with Text

* 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. Specifically, vectors of numbers.
* This is called feature extraction or feature encoding.
* A popular and simple method of feature extraction with text data is called the bag-of-words model of text.

What is a Bag-of-Words?

* A bag-of-words model, or BOW for short, is a way of extracting features from text for use in modeling, such as with machine learning algorithms.
* A bag-of-words is a representation of text that describes the occurrence of words within a document. It involves two things:

1. A vocabulary of known words.

2. 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.

Example of the Bag-of-words Model

Step 1: Collect Data

We will first preprocess the data, in order to:

* Convert text to lower case.
* Remove all non-word characters.
* Remove all punctuations.

Text->”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.

Code:

import nltk

nltk.download('punkt')

import nltk

import re

import numpy as np

from nltk.tokenize import sent\_tokenize

# execute the text here as :

text = """It was the best of times,

it was the worst of times,

it was the age of wisdom,

it was the age of foolishness,"""

dataset =sent\_tokenize(text)

for i in range(len(dataset)):

    dataset[i] = dataset[i].lower()

    dataset[i] = re.sub(r'\W', ' ', dataset[i])

  dataset[i] = re.sub(r'\s+', ' ', dataset[i])

dataset

output:

['it was the best of times it was the worst of times it was the age of wisdom it was the age of foolishness ']

Step 2 : Obtaining most frequent words in our text.

We will apply the following steps to generate our model.

* We declare a dictionary to hold our bag of words.
* Next we tokenize each sentence to words.
* Now for each word in sentence, we check if the word exists in our dictionary.
* If it does, then we increment its count by 1. If it doesn’t, we add it to our dictionary and set its count as 1.

Code:

word2count = {}

for data in dataset:

    words = nltk.word\_tokenize(data)

    for word in words:

        if word not in word2count.keys():

            word2count[word] = 1

        else:

            word2count[word] += 1

word2count

output:

{'age': 2, 'best': 1, 'foolishness': 1, 'it': 4, 'of': 4, 'the': 4, 'times': 2, 'was': 4, 'wisdom': 1,

'worst': 1}

import  heapq

freq\_words = heapq.nlargest(100, word2count, key=word2count.get)

freq\_words

output:

['it', 'was', 'the', 'of', 'times', 'age', 'best', 'worst', 'wisdom', 'foolishness']

Step 3 : Building the Bag of Words model

* In this step we construct a vector, which would tell us whether a word in each sentence is a frequent word or not. If a word in a sentence is a frequent word, we set it as 1, else we set it as 0.

As a binary vector, this would look as follows:

[1, 1, 1, 1, 1, 1, 0, 0, 0, 0]

Code:

X = []

for data in dataset:

    vector = []

    for word in freq\_words:

        if word in nltk.word\_tokenize(data):

            vector.append(1)

        else:

            vector.append(0)

    X.append(vector)

X = np.asarray(X)

Output:

array([[1, 1, 1, 1, 1, 1, 1, 1, 1, 1]])

Advantages:

* 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.

Disadvantages:

* Bag of words leads to a high dimensional feature vector due to large size of Vocabulary, V.
* Bag of words doesn’t leverage co-occurrence statistics between words. In other words, it assumes all words are independent of each other.
* It leads to a highly sparse vectors as there is nonzero value in dimensions corresponding to words that occur in the sentence.

THANK YOU.