Document text encodings:

Bag of words (BoW) encoding

BoW(w,d)=(number of times word w occurs in d)

For example we have a document:

D1: A support vector machine is the best linear classifier. For non-linear we want to use the support vector machine again but in a transformed feature space.

D2: The linear regression classifier is probably the first classification ever introduced but the support vector machine outperforms it almost all the time on real data. The reason for this is that it is less sensitive to outliers than linear regression.

D3: Speaking of outliers the 01 loss is least sensitive and can handle outliers better than any linear classifier. But the 01 loss is very hard to optimize.

We can make a word vector that counts the number of occurrences of words in documents.

Word vector = (support, vector, machine, linear, non-linear, classifier, loss, optimize, outliers)

BoW(w,D1) = (2, 2, 2, 1, 1, 1, 0, 0, 0)
BoW(w,D2) = (1, 1, 1, 2, 0, 1, 0, 0, 1)

And in this way we can also do BoW(w,D3). Now we can compare documents and use their vector form to classify them as well.

Term frequency (TF) encoding

tf(w,d)=(number of times word w occurs in d)/(total words in d)

Inverse document frequency (IDF) encoding

idf(w,D)=log((number of documents in D)/(number of documents in D that contain the word w))

TF.IDF encoding

tf(w,d,D)=tf(w,d)*idf(w,D)

Context prediction

Given a set of words in a sentence can we predict the next word? In order to solve this problem, we need a model that takes context into consideration. A simple context model would be to predict the fifth word from the first four. For example a model would be given “The cat ate the” and the prediction is “mouse”. To build such a model we need a training dataset (xi,yi) where each xi are four words of a sentence and the yi is the fifth predicted word.

How do we encode the words so we can perform machine learning? Three choices:

1. One hot encoding: high dimensional vectors where each dimension corresponds to a word
2. Label encoding: each word maps to a unique number
3. Word2vec: Use a neural network to the word from a previous one. Use the hidden layer to represent the input word.