COMP4650/6490 Document Analysis – Semester 2 / 2023

Tutorial / Lab 3

Last updated August 18, 2023

Q1: Adding More Features

When using linear regression, we must represent our input objects as a vectors. Each component of the vector is a feature which describes something about the input (e.g. for TF vectors, each component tells the model how many times a particular word occurs in the document). Is it possible that adding more features to a dataset, that is giving more information about each object to the model, causes a model to make worse predictions? Justify why you believe this is possible or not possible.

Q2: Pairwise Mutual Information Vectors

Given the following word co-occurrence matrix

	jumps	over	dog
quick	4	3	3
brown	4	6	1
fox	2	3	4

Compute the pairwise mutual information representation vector of the words "quick", "brown", "fox".

Q3: Practical Exercise

In this lab, you will build a text classification model on a provided movie review dataset. The dataset consists of 50,000 review articles written on movies in IMDb, each review is labelled with the sentiment – either positive or negative. Your text classification model will be able to infer the sentiment of a review from its text. The main goal of this question is for you to gain familiarity with the `scikit-learn`[1] machine learning package and its application to text data.

One simple approach to classifying the sentiment of documents from their text is to train a logistic regression classifier using bag of words features. This approach is relatively straightforward to implement and can be very hard to beat in practice.

You have been provided with a notebook `lab3-sparse_linear_classifier.ipynb` which loads the movie reviews and splits the data into training, validation, and testing sets. Your task is to apply logistic regression to the movie review dataset, to predict the sentiment label from the review text. To do this you will need to write a function `fit_model` that takes a set of training sentences as input, tokenizes the sentences, calculates TF vectors and then trains a logistic regression model.

(*HINT:* `CountVectorizer`*, and* `LogisticRegression` *in the* `scikit-learn` *package will be helpful for this. You should use them after reading the documentation*).

You should also implement `test_model` as it will be useful in the next part. Using `fit_model`, `test_model`, and your training and validation sets you should then search over possible values for the regularisation parameter C. It is suggested that you try $C = 3^k$ where k is an element of $\{-5, -4, ..., 0, 1, ..., 5\}$, and select based on accuracy. Though other choices are possible, you might want to test them out. Next,

[1] `https://scikit-learn.org/`

re-train your classifier using the training set concatenated with the validation set and your best C value. Evaluate the performance of your model on the test set.

Answer the following questions:

(a) What was the best performing C value?

(b) What was your final accuracy? (The accuracy of the reference solution is around 0.87)

(c) Look at the co-efficients of the logistic regression classifier (i.e. the `coef_` attribute of the logistic regression object), what words do the 5 largest (most positive) and 5 smallest (most negative) co-efficients correspond to? (List as words not their token ids, `CountVectorizer` has a `vocabluary_` attribute that may help.)

Briefly explain why these words do or do not make sense.