NLP Coursework (2) [**Reassessment**]

Dell Zhang
Birkbeck, University of London

2021/22

Part 2 of the NLP coursework is worth 10 marks.

1. **(2 marks)**

A software vendor claims that their IR system outputs the following result for a TREC query. Is there anything suspicious? Please list *all* the errors that you can find.

Ranking	Recall	Precision
1. d_8	20%	90%
2. d_{32}	60%	80%
3. d_{98}	80%	70%
4. d_{124}	60%	60%
5. d_9	80%	50%
6. d_{78}	80%	40%
7. d_{73}	80%	30%

2. **(4 marks)**

Train two models, *multinominal* Naïve Bayes and *binarized* Naïve Bayes, both with Laplace smoothing, on the following document counts for key sentiment words, with positive or negative class assigned as noted.

doc	`good`	`poor`	`great`	class
d_1	0	1	2	pos
d_2	1	3	0	neg
d_3	1	5	2	neg
d_4	0	2	0	neg
d_5	3	0	3	pos
d_6	1	1	1	neg

Use both models to assign a class (pos or neg) to this sentence d_7:

`Good cast, good acting, great music, but poor story.`

Do these two models agree or disagree? Please show your calculations in detail for their learning and prediction.

3. **(2 marks)**

Given the same training documents as in the previous question, build a Logistic Regression model for sentiment classification, using the term frequencies of those three sentiment words (`good`, `poor`, and `great`) as the three features (x_1, x_2, and x_3) respectively.

What is the final equation for the probability of a document being positive in the constructed model? How would it classify the test document d_7?

Please write a Python program to solve this problem. You do **not** need to submit your code; only the final answers are required.

4. **(2 marks)**

Given the following 3×4 term-document matrix C, perform Latent Semantic Indexing (LSI), aka Latent Semantic Analysis (LSA), using rank-2 *truncated* Singular Value Decomposition (SVD).

$$C = \begin{bmatrix} 5 & 5 & 0 & 1 \\ 4 & 5 & 0 & 0 \\ 1 & 0 & 5 & 4 \end{bmatrix}$$

What will be U_2 the truncated SVD term matrix? What will be V_2^T the truncated SVD document matrix? What will be C_2 the rank-2 approximation of the term-document matrix?

Please write a Python program to solve this problem. You do **not** need to submit your code; only the final answers are required.