DATA - X, HW 10

Dimitrios Hytiroglou

Exercise 1

For X1

P(Y=1|X1=0) = 2/3 H(Y1|X1=0) = 2/3*LOG(1/(2/3),2)+1/3*LOG(1/(1/3),2) = 0.9182958

P(Y=1|X1=1) = 2/5 H(Y1|X1=1) = 2/5*LOG(1/(2/5),2)+3/5*LOG(1/(3/5),2) = 0.9709506

H(Y|X) = 3/8 * 0.9182958 + 5/8 * 0.9709506 = 0.951205
Info Gained = 1- 0.951205 = 0.048795

For X2

P(Y=1|X2=0) = 3/4 H(Y1|X2=0) = 3/4*LOG(1/(3/4),2)+1/4*LOG(1/(1/4),2) = 0.8112781

P(Y=1|X2=1) = 1/4 H(Y1|X2=1) = 1/4*LOG(1/(1/4),2)+3/4*LOG(1/(3/4),2) = 0.8112781

H(Y|X) = 0.5*0.8112781+0.5*0.8112781 = 0.811278
Info Gained = 1 - 0.811278 = 0.188722

For X3

P(Y=1|X3=0) = 1/2 H(Y1|X3=0) = 1/2*LOG(1/(1/2),2)+1/2*LOG(1/(1/2),2) = 1
P(Y=1|X3=1) = 3/6 H(Y1|X3=1) = 3/6*LOG(1/(3/6),2)+3/6*LOG(1/(3/6),2) = 1

H(Y|X) = 2/8*1+6/8*1 = 1
Info Gained = 1 - 1 = 0

From the above it is clear that the best feature on which to perform the first split is X2, based on it providing the highest information gain.

Exercise 2

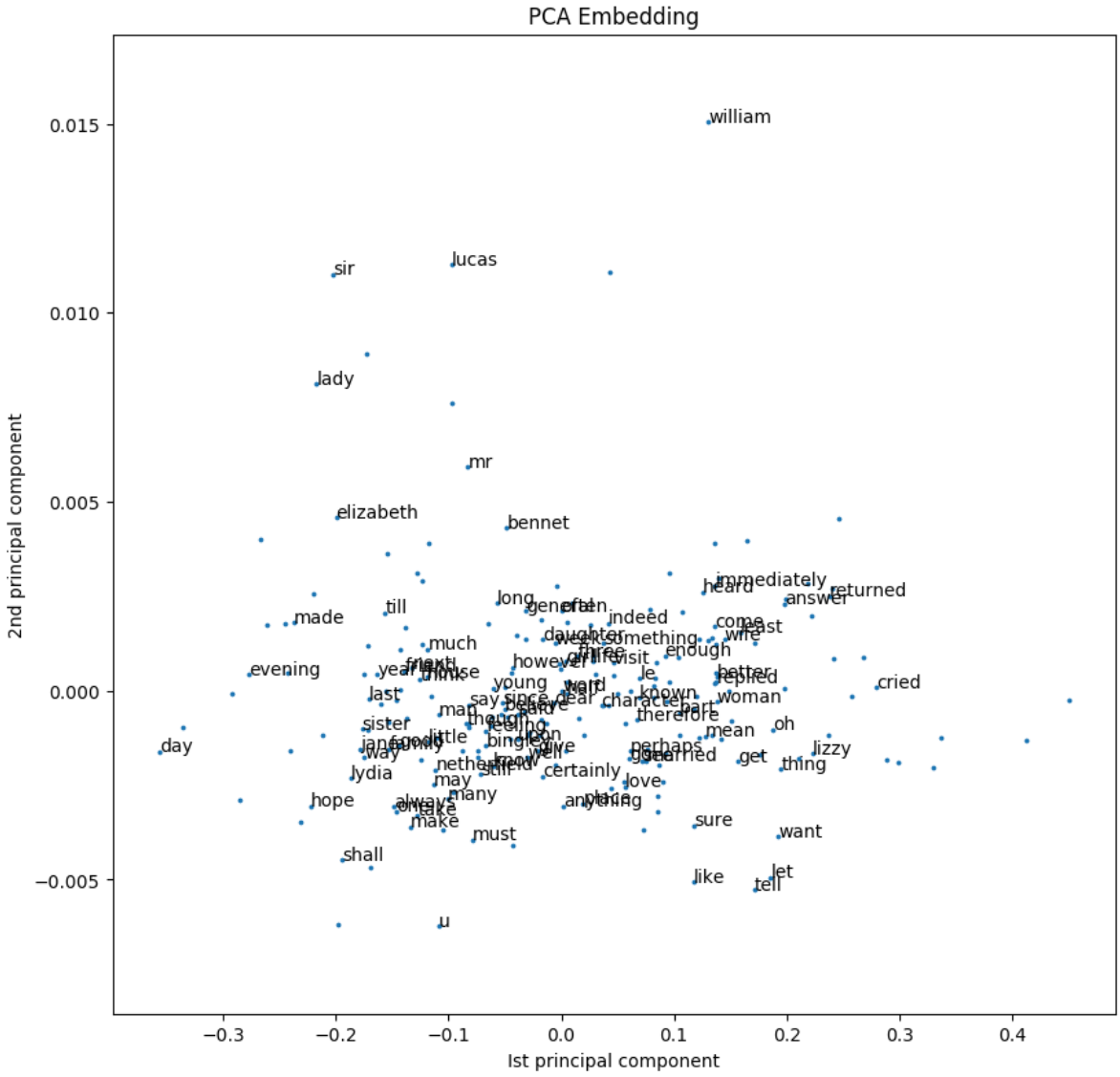
P(A) = 0.7, P(B) = 0.2, P(C) = 0.1

`H(S) =7/10*LOG(1/(7/10),2)+2/10*LOG(1/(2/10),2)+1/10*LOG(1/(1/10),2) = 1.15677965 bits`

This means that theoretically the smallest code in signal S is `1.15677965` bits per symbol.

Exercise 3

PCA decomposition of the word vectors



Vocabulary count of the model:
Vocab length: 234

Intrinsic evaluations

1)

```
print(model.similarity('elizabeth','girl'))
print(model.similarity('man','girl'))
print(model.similarity('man','carriage'))
print(model.similarity('man','opinion'))
```

```
0.99989600023
0.999891187187
0.999875494258
0.999886734582
```

2)

```
model.most_similar('kitty')
```

```
[('last', 0.9999145269393921),
 ('first', 0.9999128580093384),
 ('every', 0.9999105930328369),
 ('could', 0.9999099969863892),
 ('pleasure', 0.9999094009399414),
 ('even', 0.9999076128005981),
 ('wickham', 0.9999073147773743),
 ('said', 0.9999067187309265),
 ('thought', 0.9999065399169922),
 ('evening', 0.9999058246612549)]
```

3)

```
model.most_similar('man')
```

```
[('day', 0.9999228715896606),
 ('still', 0.9999170303344727),
 ('know', 0.999916672706604),
 ('away', 0.9999154806137085),
 ('gentleman', 0.9999138116836548),
 ('time', 0.9999130964279175),
```

```
 ('good', 0.9999123215675354),
 ('whose', 0.9999122619628906),
 ('evening', 0.9999116063117981),
 ('seen', 0.9999103546142578)]
```

4)

```
model.doesnt_match("man gentleman cousin brother aunt child kitchen".split())
```

```
'Brother'
```

5)

```
model.doesnt_match("sister brother man woman happiness carriage".split())
```

```
'woman'
```

Based on the few intrinsic evaluation tests conducted we could say that the model does not perform great. We can see, for example, that in the similarity test there is tiny if any difference in the similarity of clearly much different to each other words. Furthermore, in the “Does not match” tests, the model fails repeatedly to detect the word that indeed does not match. Finally, we can see in the “most similar” tests, the model fails to find truly similar words, which really indicates that the problem is most probably due to the very small vocabulary size.