Foundations of machine Learning II

Project: Compression, Prediction, Generation Text Entropy

Flavie Vampouille

MSc in DS&BA

ESSEC Business School & CentralSupélec

Questions

1. Interpret the time invariant assumption associated to our Markov chains for text compression.

In text compression the usual time indices which indicate the place in the time line are replaced by indices that indicate the position in the text. Hence the time invariant assumption associated to our Markov chains for text compression means that for sequences of characters $(X_i)_i$ we have

$$P(X_i|X_{i-1}) = P(X_2|X_1) \quad \text{for all } i$$

That means that whichever is the place of the character in the text, the conditional probability of the next characters is the same.

2. How can we rewrite a Markov chain of higher order as a Markov chain of order 1?

A Markov chain of order k satisfies

$$P(X_i|X_{i-1}, \dots, X_1) = P(X_i|X_{i-1}, \dots, X_{i-k})$$

While a Markov chain of order 1 satisfies

$$P(X_i|X_{i-1}, \dots, X_1) = P(X_i|X_{i-1})$$

In the case of text compression we can understand than while the order 1 means that we are studying a character of length 1, in the case of order k we are studying symbols which contain each k characters.

3. Given a probability distribution over symbols, how to use it for generating sentences?

Given a probability distribution over symbols, we can use it to generate fake sentences.

In IIDModel:

1. Choose the order that minimize the cross entropy and hence would give the more similar sentences that the language we are studying.
2. Pick randomly symbols with respect to their frequencies distribution in the training text.
3. Concatenate these symbols to obtain a fake sentence.

In MarkovModel:

1. Choose the order that minimize the cross entropy and hence would give the more similar sentences that the language we are studying.
2. Given a first k order symbol choose randomly or with other methods, look at the characters directly after this symbol and choose one with respect to the frequencies. Add it at the end of the symbol and do it again after dropping the first character…

4. As for supervised learning, a model can overfit the training set. Propose a simple approach (cost function) for measuring the goodness of the model.

As for supervised learning, to avoid overfitting we can define a cost function that is based both error (here the entropy: E) and a penalization parameter (here the order of the model is k)

$$Cost = H + \alpha.k$$

that we want to minimize. We usually do that by cross validation.

Implementation

1. For different orders of dependencies, train the model on a novel and compute the associated entropy. What do you observe as the order increases? Explain your observations.

I run the entropy in IIDModel and MarkovModel with different order values and obtain the following results:

order	1	5	100	1000
IIDModel entropy	4.426	14.303	20.103	20.101
MarkovModel entropy	3.483	1.369	0.0	0.0

IIDModel

We can see that when the order (number of characters in the symbol) increases the entropy increases. This is due to the fact that the number of possible symbols increase: for order 1 we have basically the letters (lower and upper case) the blank and the non-letters signs; for order two we have combination of pairs of letters and signs… Each time the order increases the number of possible symbols increases and so does the entropy of the text. However, if the order becomes big, we will observe a new decrease in the entropy as there will be fewer possible symbols due to the limitation of the text size (example: if the text is 123456 and the order is 6 we will have only one symbol: 123456 and thus the entropy will be zero).

Note: As for order 1 I considered each symbol independently, I do the same for the others orders. For example for order 2 if the text is 123456 I consider the pairs 12 23 34 45 56. They are overlapping but I choose to put every possible contiguous pair in my dictionary.

MarkovModel

The entropy decreases as the order increases and tends toward zero. Once again because of the finite size of the text, big order value only conduct to low information and if the order is to big the "next character" is entirely determined by the previous symbol and there is no more different probabilities.

2. Use the other novels as test sets and compute the cross-entropy for each model trained previouly. How to handle symbols (or sequences of symbols) not seen in the training set?

When computing the cross-entropy of two files a new issue appears: some symbols x are not in both files, so the sizes of the frequencies vectors p and q are not the same and the code crash due to dimensionality incompatibility in the formula of the cross entropy.

On my first part of code I use a dictionary to register the number of occurrences of each symbol and the just stored the frequencies in a vector with no key values.

However, if it computes the entropy, this doesn't allow me to compare the symbols' occurrences in both text nor to be sure that the frequencies I stored for both text are in the same order.

I then store the frequencies in dictionaries, compare the two dictionaries and use the key value to compute the cross entropy

In IIDModel with

- The usual formula $p(x).\log 2\, q(x)$ if the key is in both dictionaries (where q is the distribution in the target text and p the distribution in the source text).
- $p(x).log2\frac{1}{lengt\_dictionary\_target}$ if the key appears only in the source
- $\frac{1}{length\_dictionary\_source}.log2\ q(x)$ if the key appears only in the target

In MarkovModel with

- With $\sum_x (p(x) * \sum_y(p(y|x) * log2\ q(y|x)))$ if the symbol is on both dictionaries
- $\sum_x (p(x) * \sum_y(p(y|x) * log2\ \frac{1}{lengt\_dictionary\_target}))$ if the key appears only in the source
- $\sum_x (\frac{1}{length\_dictionary\_source} * \sum_y(\frac{1}{length\_dictionary\_source} * log2\ q(y|x)))$ if the key appears only in the target

3. For each order of dependencies, compare the cross-entropy with the entropy. Explain and interpret the differences.

I run the code for several orders with 'Dostoevsky.txt' as text source and the three others as target and obtain these results below. I run the cross entropy after order 4 for some texts and see that it always. I report it for the first four orders as they are enough to see what happens.

SCR text entropy

Order	1	2	3	4
IIDModel entropy	4.4267	7.9106	10.5969	12.6468
MarkovModel entropy	3.4839	2.6863	2.0499	1.6569

With "Goethe.txt"

Order	1	2	3	4
IIDModel cross entropy	5.9051	13.2963	18.5882	21.6468
MarkovModel cross entropy	5.7392	4.2415	5.0462	9.0200

With "Hamlet.txt"

Order	1	2	3	4
IIDModel cross entropy	5.6756	12.7111	18.3659	21.3285
MarkovModel cross entropy	6.1659	3.9515	4.7406	8.7933

With "Alighieri.txt"

Order	1	2	3	4
IIDModel cross entropy	7.2637	15.7081	19.3784	24.3571
MarkovModel cross entropy	5.6337	5.3511	6.9735	10.5935

The cross-entropy is always bigger than the entropy, which could be expected as the Kullback-Leibler divergence is always positive.

We can see that for texts in close languages the entropy and cross-entropy are closer and that the gap between these two values gets bigger when the order increases. This could be expected as text in language with same roots will have more symbols in common but the differences between symbols will be bigger if they have more caracters.

4. Choose the order of dependencies with the lowest cross-entropy and generate some sentences.

The best order for IIDModel is given by the lowest cross entropy. Hence to generate sentences I use the order of minimum entropy I obtained which is 1 for IIDModel and 2 for MarkovModel.

With IID Model, order 1, length 1 000, scr = Dostoevsky

nargeiwrtavdbp rldasee ofeolh,. hi aro cpelaz'goe m"ets dchgp v ph, eSa, urn a fagnsyaltlw"yiad ,ha ttfcdw ot ednghe.agaae id oeuagro iritenteRi iad eduat.uls sol i"dnpp ou nhtehi tas ide,sant basn ehllo ou ryoh nhksaooprrwr u e adlefiplluo k rvnbctenrnc t tcalh.rsd.ctir oteowat ! dlesotthnLoh ,dirloeoiwapsdInil "i okesoHdslwn' tl .b oetnbgheo,Yehnmt ut,eacir t r,dg wstwangyu mc'e?scfArnycdsdoktve rfgccdat rayn heeeu gg.se iraln aohy ydd tteasthi u ha our. nefaitwMaegPocdCtor flgaaee amhWtmm nurrk ec vrnrme wtfrtyripntmraav?g wh oohw eowtnss'nt htaatnlAe if?anhv o" tttes ln.atdhstao n hh lw syrpdnsS.rl e ee?n i atftntcew,pla ilegeoIeeuot ts u,t.whl mha.ttisbhn vdgtr sepi 'ef iaorntt upra ti,aleaee.eehsnn (ett fh lsdtst a ba neHiysot latbyetepaoo s iswotui,vHk midn trsdhwi shcnyeuaotR eent,.'ftoIgoaeeoueea.ta'hiahohe o hmwyNrhtit pt lh wiyh r nsoiamh,lyiayephP trsRhgs ia teea i, lseiwtc,oipwbfoiaio lt, tte caf fiodt eiaaa drror gag l eaonn ehuib de frhohgi rio eseto

With MarkovModel, order 2, length 1 000, scr = Dostoevsky

card non mom nia nothe covia lood!" "examy anove whe twor?" com tookin.. Why, ov's so told nowe thed. He youlgan....." "I be thou, frove nove he hunkintrat. Where rimes he Mikold rosea I and some samessim Aman to ven thim. *** "Whall hagait's suffelin ress). Is of the Pyouthe likoldefuld hat there hing therem," hing at I? Onest withere wite man antle whed..... No, hen therdsore know morst had, "thin herembefortuniatim a go go gly. alwashunter ang a beford to him was driagetch whar he custry, to up. I donsim I dood of an aftled. "The a be red the hat anter mom to of speconot led nall!" "Why, begait advit, agaperepead sed Dagaince"? Wel the've fal, quiche the onglas as bantent ber put on was alwas he de was a halin, me compablovna lar juserwaskult of the ken askold itcharsting ing be es. I sortyart wit that's tand thadeen dreyescreas ad and maying ov bar st som an't she sin al plas againg the is squaid king a samit pre, ye. I cout," "Haventick aris oned way twout or murgrelf, do se

We can see that the text generated with MarkovModel looks more than English that the one generated with IIDModel.

5. Train one model per novel and use the KL divergence in order to cluster the novels.

KL divergence = $\sum_x p(x).\log 2 \frac{p(x)}{q(x)}$

= cross-entropy (p,q) - entropy(p) with IIDModel

We have at order 1 (we use order 1 because it gives the smallest cross-entropy with IIDModel for every text association):

source	**target**	**KL divergence**
Dostoevsky	Goethe	1.4783
Dostoevsky	Hamlet	1.2488
Dostoevsky	Alighieri	2.8369
Goethe	Dostoevsky	4.4919
Hamlet	Dostoevsky	5.1804
Alighieri	Dostoevsky	5.0708
Goethe	Hamlet	1.6132
Goethe	Alighieri	2.9833
Hamlet	Goethe	2.2312
Alighieri	Goethe	2.3114
Hamlet	Alighieri	3.7782
Alighieri	Hamlet	2.6298

As KL divergence is not symmetric I use d(p,d) = $\frac{KL(p||q)+KL(q||p)}{2}$ as a metric.

d (Dostoevsky , Goethe)	2.9851
d (Dostoevsky , Hamlet)	3.2146
d (Dostoevsky , Alighieri)	3.9538
d (Hamlet , Goethe)	1.9222
d (Hamlet , Alighieri)	3.204
d (Goethe , Alighieri)	2.6473

The closest texts are Hamlet and Goethe. As I find that really unexpected I take a look at the txt files and it appears that both hamlet and Goethe are written in German, while Dostoevsky is in English and Alighieri in Italian. Hence it is logical than Hamlet and Goethe are the closest.

We have

$$d(\text{Hamlet}, \text{Goethe}) < d(\text{Goethe}, \text{Alighieri}) < d(\text{Dostoevsky}, \text{Goethe}) < d(\text{Hamlet}, \text{Alighieri}) < d(\text{Dostoevsky}, \text{Hamlet}) < d(\text{Dostoevsky}, \text{Alighieri})$$