Concept of KL divergence

In any business problem, After finding "what variables are", natural question following is "how to use them". With sentiment analysis, once class of sentiment is known [broadly – Positive, Negative, Neutral]; one needs to know how to use this knowledge. One novel application can be - using change in relative probability distribution of data against target.

Quantification of these sentiments give confidence in sentiment – closer the samples, stronger the sentiment. One approach is to calculate distance measure between the two distributions. This is challenging for text analytics as interpret of the measures is often difficult.

Instead, divergence [ref] between two probability distributions can be calculated. A divergence is measure, but not a metric as it is not symmetrical. This means that a divergence is a scoring of how one distribution differs from another. Consider two probability distributions P and Q. Usually, P represents the data, the observations, or a probability distribution precisely measured. Distribution Q represents instead a theory, a model, a description or an approximation of P. The **Kaulback–Leibler divergence** is then interpreted as the average difference of the number of bits required for encoding samples of P using a code optimized for Q rather than one optimized for P. [ref]

$$D_{KL}(P \,||\, Q) = \sum_{x \in X} P(x) \log \frac{P(x)}{Q(x)}$$

Now applying to a real-world Data - Suppose, we have 2 statements from random Opinionator which both are "Negative":

- As both belong to same class, they necessarily convey SAME sentiment.
- Statement1 is considered an "observation" of class Negative
- Statement2 can be considered an "Approximation" of Statement1, as it also represents class "Negative"
- Hence, for single random variable expressing sentiment "Negative" – we have two different probability distributions for the variable, such as true distribution and approximation of that distribution.

So, in above example, discrimination between observation and approximation(which can be simply replaced by ML prediction model) is identified using KL divergence.

Assuming normal distribution of dataset, KL-distributions amongst random variable sets [Positive, Negative, Neutral] should typically move symmetrically. When there is distinctive difference i.e. systematic inverse proportion between 2 variables; one can say EVENT HAS OCCURRED.

In following discussion, terms (observation, model) are used when referencing relation of form $D_{KL}(P,Q)$

1 a) Take the simple sentences and put these into the program to compute the K-L divergence scores for them, in both directions.

The two strings used to calculate the KL Divergence are as follows:

- statement 1: john fell down harry fell as-well down by the stream the sun shone before it went down mary was fine
- statement 2: bill fell down jeff fell too down by the river the sun shone until it sunk down belinda was ill

To find KL divergence between 2 statements, KL Divergence Algorithm ingests them to determine the score between the two statements. Algorithm for KL difference between S1 and S2 is stated bellow:

- stop words are removed from Input statement 1 and 2. Also, any words smaller than 3 characters are omitted. Then, remaining corpus is tokenised.
- Words present in the string 1 but not in 2 are stored in list VocabDiff.
- To determine the KL Divergence between statement 1 and statement 2; relative probability distribution is considered along with the parameters such as Epsilon and Gamma. Epsilon is the probability of a rarity [term with least term frequency]. Gamma normalizes the statement 2 probability.

```
print("KL-divergence between d1 and d2:", kldiv(tokenize(d1), tokenize(d2)))
print("KL-divergence between d2 and d1:", kldiv(tokenize(d2), tokenize(d1)))
```

Tokenization output – term frequency:
Statement 1: {'john': 1, 'fell': 2, 'down': 3, 'harry': 1, 'well': 1, 'by': 1, 'stream': 1, 'sun': 1, 'shone': 1, 'before': 1, 'went': 1, 'mary': 1, 'was': 1, 'fine': 1} **COUNT : 14**
Statement 2: {'bill': 1, 'fell': 2, 'down': 3, 'jeff': 1, 'by': 1, 'river': 1, 'sun': 1, 'shone': 1, 'until': 1, 'sunk': 1, 'belinda': 1, 'was': 1, 'ill': 1} **COUNT : 13**

Divergence output:
- **KL-divergence between S1 and S2: 3.24943**

Statement1 diff staement2 : {'mary', 'stream', 'well', 'john', 'harry', 'went', 'before', 'fine'}
- **KL-divergence between S2 and S1: 3.04783**

Statement2 diff staement1: {'bill', 'river', 'until', 'sunk', 'jeff', 'belinda', 'ill'}

Observations:
- D(S1-S2) > S(S2-S1) : As seen, S1 contains more unique words which are not in S2 (S1-S2) as against unique words in S2 which are not in S1 (S2-S1).
- **When S1 is considered a reference model statement; and it is used to describe (or encode) S2; it does better; as against S2 describing S1.**
- **Higher dissimilarity between observation and model increases DL score.**

1 b) Now create a third story that is very different to the other two, add it to the program.

User input taken is : 'omeara is nice elizabeth approved them kennedy is mystery man'
Tokenization output – term frequency:
Statement 3: {'omeara': 1, 'nice': 1, 'elizabeth': 1, 'approved': 1, 'them': 1, 'kennedy': 1, 'mystery': 1, 'man': 1, 'sun': 1, 'sets': 1, 'on': 1, 'britanias': 1, 'waves': 1} **COUNT : 13**

KL-divergence between S1 and S3: 6.77627
Statement1 diff staement3 : {'john', 'by', 'mary', 'stream', 'well', 'harry', 'down', 'was', 'shone', 'fell', 'went', 'before', 'fine'}
KL-divergence between S3 and S1: 6.62383
Statement3 diff staement1 : {'approved', 'nice', 'kennedy', 'on', 'man', 'britanias', 'them', 'mystery', 'elizabeth', 'sets', 'waves', 'omeara'}

KL-divergence between S2 and S3: 6.623832
Statement2 diff staement3 : {'by', 'river', 'belinda', 'sunk', 'was', 'ill', 'shone', 'fell', 'jeff', 'down', 'bill', 'until'}
KL-divergence between S3 and S2: 6.565735
Statement3 diff staement2: {'approved', 'nice', 'kennedy', 'on', 'man', 'britanias', 'them', 'mystery', 'elizabeth', 'sets', 'waves', 'omeara'}

1 c) Report how its score changes relative to the first two. Comment on whether the scores make sense.

Summary of observations from 1 a) and 1 b) is tabled bellow:

Statement Pair	Count(different terms)	Epsilon (e-05)	Gamma	KL-divergence
S1 - S2	8	5.8823529412	0.9995294118	3.2494321223
S2 - S1	7	5.8823529412	0.9995882353	3.0478297684
S1 - S3	13	5.8823529412	0.9992352941	6.7762742938
S3 - S1	12	5.8823529412	0.9992941176	6.6238317640
S2 - S3	12	6.2500000000	0.9995000000	6.6238317640
S3 - S2	12	6.2500000000	0.9995000000	6.5657354364

Obtained results are consistent with previous observations - **Higher dissimilarity between observation and model increases DL score.**

- S2 and S2 both have 13 unique terms which are not present in one another. But, S2 has few terms with higher term frequency ['fell': 2, 'down': 3]. Hence, D(S2-S3) > S(S3-S2)
- Statement 3 is completely district snippet and has very small overlapping terms compared to overlap between S1 and S2
- As seen, epsilon for S1 with (S2,S3) is same. This is because Epsilon depends on value of rarity term divided by sum of occurrences of words. S1 has higher vocabulary length, so it dominates the parameter.

1 d) Explain what role *epsilon* and *gamma* play in the computation of K-L.

```
""" epsilon """
epsilon = min(min(_s.values())/ssum, min(_t.values())/tsum) * 0.001

""" gamma """
gamma = 1 - lenvocabdiff * epsilon
```

Epsilon is the back-off probability of a rarity [term with least term frequency]. In KL divergence calculation, the term probability distribution of a document is compared with each category probability distribution [ref]. "Back-off model" discounts the term frequencies appearing in the document and terms which are not in the document are given an epsilon probability equal to the probability of unknow words. For the terms not in document, it is useful to introduce a back-off probability. Otherwise the distance measure will be infinite.
Gamma is normalizing coefficient [ref] which accounts the weight introduced by **epsilon**, thus making sure that the probability of the term in the category satisfies the property of the probability to be one and makes sure it sums to 1.

From code, we can see that:

- Epsilon assigns a minimal probability to terms not present in model document; which is lesser than minimum term probability in both observation and model. When term in observation is absent in model, it is assigned minimal probability epsilon.
- When term in observation is present in model, it is assigned normalized probability gamma.