Lecture NO.1

Language

- Formal
  - C++
  - Python
  - Assembly
  - etc
- Informal
  - Human Language.

- NLP Processing:
  - → Acquisition (Sources: Speech, Text etc)
  - → Representation (meaningful)
  - → Understanding
  - → Tockenization etc.

- Applications of NLP
  - → Machnice Translation
  - → Sentiment Analysis
  - → Language Technology

- Regular expressions are used when there are infinite / unlimited sets. Limited sets are saved.

Regular Expressions

- Regular Operands
- Regular Operators

[Ii]|[Tt]

Lecture NO.2

Regular Expression → used in string matching, finding, tokenization etc.

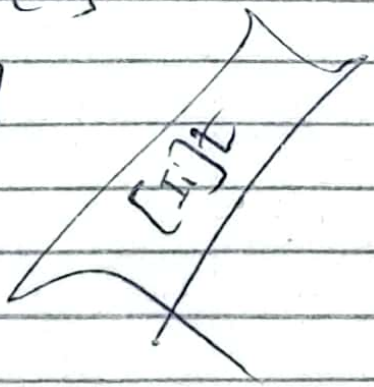
→ Operations:

? (0/1 occurence) → for specific character
* (0/N occurences)
+ (1/N occurences)
. (for any character)
^[] (start of line)
$ (end of line)
\. (character '.')
. (for any character)

[Ii][Tt] [Ii]

[Ii][Tt]

[Ii][Tt]



·) decrease in false positive, increase in precision

·) Recall

·) FP (Type I error)

→ 'V' → vocabulary → set of unique elements (unique)

→ tokens → all occurences (redundant + unique)

→ no. of elements / size

·) |V| → cardinality of set
N → Total occurrences

→ Byte Pair Encoding → learning based
(segmentation of words)

	Frequency	Train	Test
low_	5		new_ ✓
lowest_	2		newer_ ✓
newer_	6] 9		lower_
wider_	3]		

lower_ → low, er_ → 2 tokens

Vocabulary: (characters)
o, w, l, n, r, er, er_, ...

·) not ~~a~~ the best approach but better than tokenization based on space.

→ extrinsic model → need some real life experimentation. needs a classifier

→ intrinsic model → do not need any classifier.

Lecture NO. 3+4

(grammar) Language Models → use probabilities (conditional)

- Machine Translation
- Spell correction
- Speech Recoginition

→ Markov Assumption

(Look at a local window) — can limit the size of context

cannot increase the size so much

$LW = \{0, 1, 2, 3, \dots\}$

→ Unigram is not a good model as there is a possibility of it giving a word without any knowledge of context.

$$P(\text{water}) = \frac{C(\text{water})}{N}$$

→ Perplexity: → intrinsic model

$$\left(P(w_1, w_2, w_3, w_4, \dots, w_n)\right)^{-\frac{1}{N}} \cdot \left(\frac{1}{P(w_1, w_2, w_3, \dots, w_n)}\right)^{1/N}$$

$$= \sqrt[N]{\frac{1}{P(w_1, w_2, w_3, \dots, w_n)}} = \sqrt[N]{\frac{1}{\prod P(w_i | w_{i-1})}}$$

↓ product of

unaware of context

Activity-2

unigram:

P(I want to eat Turkish Pizza) =

$$P(I) \times P(want) \times P(to) \times P(eat) \times P(Turkish) \times P(Pizza)$$

$$= \frac{7}{44} \times \frac{7}{44} \times \frac{7}{44} \times \frac{6}{44} \times \frac{1}{44} \times \frac{3}{44}$$

$$= 0.15 \times 0.15 \times 0.15 \times 0.13 \times 0.02 \times 0.06$$

$$= 0.00000053$$

P(I want to eat Indian food) =

$$P(I) \times P(want) \times P(to) \times P(eat) \times P(Indian) \times P(food)$$

$$= \frac{7}{44} \times \frac{7}{44} \times \frac{7}{44} \times \frac{6}{44} \times \frac{4}{44}$$

$$= \frac{7}{0.15} \times 0.15 \times 0.15 \times 0.13 \times 0.02 \times 0.09$$

$$= 0.00000078$$

$$= \log\frac{7}{44} + \log\frac{7}{44} + \log\frac{7}{44} + \log\frac{6}{44} + \log\frac{0}{44} + \log\frac{4}{44}$$

$$= -4.31$$

Bigram:

P(I want to eat Turkish food) =

$$P(I|<s>) \times P(want|I) \times P(to|want) \times P(eat|to) \times P(Turkish|eat) \times P(food|Turkish) \times P(</s>|food)$$

- Perplexity: smaller the perplexity, higher the confidence of model
  - If 3 words have same probabilities then the perplexity will also be 3
  - If 10 words have same probabilities then the perplexity will be 10.

Ex:

$$\sqrt[3]{\frac{1}{0.7 \times 0.15 \times 0.15}} = 0.85$$

0.7 → w_1, 0.15 → w_2, 0.15 → w_3

The perplexity will be less than the number of words if the probabilities are not same.

Ex:

$$\sqrt[3]{\frac{1}{1/3 \times 1/3 \times 1/3}} = \sqrt[3]{\frac{1}{1/27}} = 3$$

- smaller the perplexity, better the model but ~~sometimes~~ also depends on vocabulary.

Lecture NO·5

→ Closed Vocabulary:
Vocab that is already defined, no more new words can be added

→ After applying log when the prob is 0 of any world then the answer is in log space and might be negative

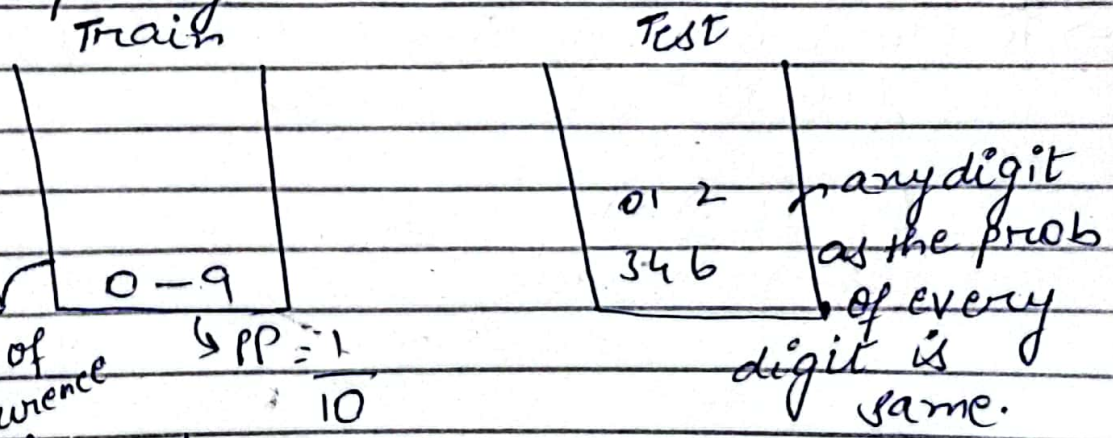
$\log(P_1 \times P_2) = \log_2 P_1 + \log_2 P_2 = x$

(TO convert to probability 2^x space again)

we convert answer into log space to avoid underflow.

→ Handle the context properly & efficiently.

Ex:
Perplexity



OOV → out of vocabulary

•) Ex → PP = 3 → confused b/w 3 words.

Train

0.90	0000
0.05	1000
0.05	2000
	1000

Test

Ex1	000	max prob that it will predict 0
Ex2	0011 22	→ PP = 3 confused b/w all words

→ Laplacian Smoothing:

Disadvantages:

→ increase in perplexity

→ possibility that the probability of words that were not in the corpus may be greater than the prob of words that already exists in trainning set/corpus.

$$\frac{C(w_{i-1}, w_i) + 1}{C(w_{i-1}) + |V|}$$

→ Add K-Smoothing (varient of Laplace) (not a good approach)

$$\frac{C(w_{i-1} w_i) + K}{C(w_{i-1}) + |V| \times K} \quad \text{OR}$$

~~$C(w_i, w$~~

→ Linear Interpolation:
mix of unigram, bigram and trigram.

Training Data	Held Out Data	Test Data

→ Stupid Backoff
(only good for web)

-) trigram → available (pick that)
-) if trigram not available go to bigram
-) if bigram not available go to unigram.

→ Good Turning Smoothing:

Ex:
10 carp, 3 perch, 2 white fish,
1 trout, 1 salmon, 1 eel = 18 fish
(rare)

$N_1 = \frac{3}{18}$ (add rare) N = no. of fishes (classes)

We will relate unseen data using rare data's frequency / probability.

$$c^* = \frac{(c+1)N_{c+1}}{N_c}$$

$C^*(\text{trout}) = 2 * N2/N1$

$= 2 * 1/3$

$= 2/3$

$P^*_{GT}(\text{trout}) = C^*/N = 2/3/18 = 1/27.$

Ex2:

10 carp, 3 perch, 2 whitefish, 2 trout, 2 salmon, 2 eel = 21 fish

$N0 = 0$ $\quad N1 = 0$

$N2 = 4/21$ $\quad N3 = 1/21$

$C = 2$

$C^*\text{trout} = 3 \times 4/21 = 4/7 = 2/21 = 3 \times 2/7$

~~$P^*(\text{trout}) =$~~

$P^*_{GT} = N2/N = 4/21$

Prob of unseen

$C^*(\text{trout}) = (2+1) \times N3/N2$

$= 3/4$

$P^1(\text{trout}) = 3/4/21 = 1/28$

Regular Expressions

Morphology

- Stemming
- Lematization

> used for text normalization

> used for decreasing length of vocabulary and computational cost.

Stemming: chop off

maps all variations of 1 words on a stem

→ Stem: core meaning bearing morphemes

Ex: cats
Stem: cat
affix = s

) used in search engines
) used for info retrieval

→ In machine translation, lemmatization is better approach but increase computational cost.

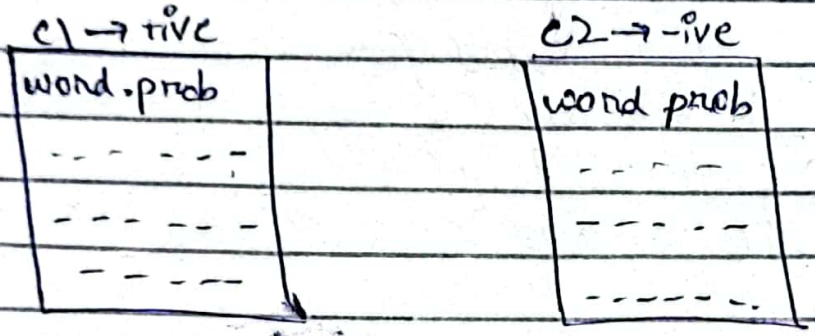
Lecture NO: 6

Classification

- Probabilistic models

parameter → V

V → c (class)



- "dictionary" → used to store it

Naive Bayes

First, we

- have to convert the corpus in "bag of words"

- argmax [P(c1|d), P(c2,d)]

↓

will return the class with highest probabilities. We can add many values (classes) in it.

Activity-1

Prior Probabilities =

$c = \frac{3}{4}$ $\quad$ $j = \frac{1}{4}$

$c = 0.75$ $\quad$ $j = 0.25$

Vocabulary = Chinese, Beijing, Shanghai, Macao, Tokyo, Japan

Vocab size = 6

$MD_c = d_1 + d_2 + d_3$ $\quad$ $MD_c(Vocab) = 5$

$MD_j = d_4$ $\quad$ $MD_j(Vocab) = 3$.

$P(China|c) = \frac{5}{8}$ → words in class c

{c: {China: 5/8 .}, ... } j: { } }