Haramaya University

College of computing and informatics

Department of Computer Science

Natural Language Processing

Assignment

Submitted by: Abrahim Abdo

ID:pgp/ 768/13

Submitted to: Dr.Worku J

Submission date: February 2, 2022

2. Collect adjectives and adverbs as many as possible from online. Divide the dataset into two separate folders (70% for training and 30 for testing), then train SVM with the collected dataset.Evaluate the accuracy of the model? Add the number of dataset again and train the model? Present what differences you have observed? In case the class is imbalanced, use data augmentation so that the two class are equiprobable. Is there any accuracy difference for the imbalanced classes and after the class is balanced through data augmentation?

Yes ,there's a little difference

Without balanced, we get the fllowing accuracy

```
svcl = svm.SVC(kernel="linear")
```

Results for Support Vector Machine with tfidf
0.5014765100671141

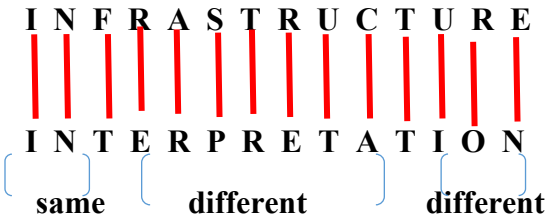
With balanced, we get the fllowing accuracy

```
pipeline_svm = make_pipeline(vectorizer,  SVC(probability=True, kernel="linear",
class_weight="balanced"))

grid_svm = GridSearchCV(pipeline_svm,

                param_grid = {'svc__C': [0.01, 0.1, 1]}, cv = kfolds,

                scoring="roc_auc",

                verbose=1,    n_jobs=-1)
```

0.5133505329140227

3. Calculate the minimum edit distance between the following words:

a. infrastructure and interpretation



Step 1: Draw the edit distance matrix. In this, each word is preceded by # symbol which marks the empty string.

	#	i	n	F	r	a	s	t	r	u	c	t	u	r	e
#															
i															

n															
t															
e															
r															
p															
r															
e															
t															
a															
t															
i															
o															
n															

Step 2: Find the edit-distance values using minimum edit distance algorithm to convert # (row side) to #infrastructure and populate appropriate cells with the calculated distance.

	#	i	n	F	r	a	s	t	r	u	c	t	u	r	e
#	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
i															
n															
t															
e															
r															
p															
r															
e															
t															
a															
t															
i															
o															
n															

Step 3: Find the edit-distance values using minimum edit distance algorithm to convert # (column side) to #interpretation and populate appropriate cells with the calculated distance.

	#	i	n	f	r	a	s	t	r	u	c	t	u	r	e
#	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
i	1														
n	2														
t	3														
e	4														
r	5														
p	6														
r	7														
e	8														
t	9														
a	10														
t	11														

i	12														
o	13														
n	14														

Step 4: From this step onwards, we try to find the cost for a sub-problem by finding the minimum cost of three sub-problems and add 1 with that if the characters intersect at that cell are different. If the intersecting characters are same, then we add 0 with the diagonal cell value.

	#	i	n	f	r	a	s	t	r	u	c	t	u	r	e
#	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
i	1	0	1	2	3	4	5	6	7	8	9	10	11	12	13
n	2	1	0	1	2	3	4	5	6	7	8	9	10	11	12
t	3	2	1	1	2	3	4	4	5	6	7	7	8	9	10
e	4	3	2	2	2	3	4	5	5	6	7	8	8	9	9
r	5	4	3	3	2	3	4	5	5	6	7	8	9	8	9
p	6	5	4	4	3	3	4	5	6	6	7	8	9	9	9
r	7	6	5	5	3	4	4	5	5	6	7	8	9	9	10
e	8	7	6	6	4	4	5	5	6	6	7	8	9	10	9
t	9	8	7	7	5	5	5	5	6	7	7	7	8	9	10
a	10	9	8	8	6	5	6	6	6	7	8	8	8	9	10
t	11	10	9	9	7	6	6	6	7	7	8	8	9	9	10
i	12	10	10	10	8	7	7	7	7	8	8	9	9	10	10
o	13	11	11	11	9	8	8	8	8	8	9	9	10	10	11
n	14	12	11	12	10	9	9	9	9	9	9	10	10	11	11

b. communications and Interpretation

Step 1: Draw the edit distance matrix. In this, each word is preceded by # symbol, which marks the empty string.

	#	c	o	m	m	u	n	i	c	a	t	i	o	n	s
#															
i															
n															
t															
e															
r															
p															
r															
e															

t															
a															
t															
i															
o															
n															

Step 2: Find the edit-distance values using minimum edit distance algorithm to convert # (row side) to #communications and populate appropriate cells with the calculated distance.

	#	c	o	m	m	u	n	i	c	a	t	i	o	n	s
#	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
i															
n															
t															
e															
r															
p															
r															
e															
t															
a															
t															
i															
o															
n															

Step 3: Find the edit-distance values using minimum edit distance algorithm to convert # (column side) to #interpretations and populate appropriate cells with the calculated distance.

	#	c	o	m	m	u	n	i	c	a	t	i	o	n	s
#	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
i	1														
n	2														
t	3														
e	4														
r	5														
p	6														
r	7														
e	8														
t	9														
a	10														

t	11														
i	12														
o	13														
n	14														

Step 4: From this step onwards, we try to find the cost for a sub-problem by finding the minimum cost of three sub-problems and add 1 with that if the characters intersect at that cell are different. If the intersecting characters are same, then we add 0 with the diagonal cell value.

	#	c	o	m	m	u	n	i	c	a	t	i	o	n	s
#	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
i	1	1	2	3	4	5	6	6	7	8	9	9	10	11	12
n	2	2	2	3	4	5	5	6	7	8	9	10	10	10	11
t	3	3	3	3	4	5	6	6	7	8	8	9	10	11	11
e	4	4	4	4	4	5	6	7	7	8	9	9	10	11	12
r	5	5	5	5	5	5	6	7	8	8	9	10	10	11	12
p	6	6	6	6	6	6	6	7	8	9	9	10	11	11	12
r	7	7	7	7	7	7	7	7	8	9	10	10	11	12	12
e	8	8	8	8	8	8	8	8	8	9	10	11	11	1`2	13
t	9	9	9	9	9	9	9	9	9	9	9	10	11	12	12
a	10	10	10	10	10	10	10	10	10	9	10	10	11	12	13
t	11	11	11	11	11	11	11	11	11	10	9	10	11	12	13
i	12	12	12	12	12	12	12	11	12	11	10	9	10	11	12
o	13	13	12	13	13	13	13	12	12	12	11	10	9	10	11
n	14	14	13	13	14	14	13	13	13	13	12	11	10	9	10

c. both a and b has 14 words? If the minimum edit distance of a and b differs, explain why it happened?

- The alike of two words are far apart
- The internal element of words are not similar
- They have no much word to each other, that is similar to one another

4. Write a brief report on Regular Expression and information extraction?

- **A formal language is an abstraction of the general characteristics of programming languages.**

It consist of a set of symbols and some rules of formation by which those symbols can be combined into entities called sentences .

- Regular Expressions and Finite State Automata
- Finite State Automata (FSA)
    - Describing patterns with graphs
      Programs that keep track of state
- Regular Expressions (RE)
    - Describing patterns with regular expressions
    - Converting regular expressions to programs
- The languages (Regular Languages) recognized by FSA and generated by RE are the same
- There are languages generated by grammars that are not Regular
- A language L is called regular if and only if there exist some deterministic finite accepter M such that
  L =L (M)
- Every regular language can described by some
- DFA
- NFA

- A deterministic finite accepter or DFA is defined by the quintuple
  $M=(Q,\Sigma,\delta,q_o,F)$
    - ✓ Q is finite state
    - ✓ Σ is input symbol
    - ✓ δ :$Q^* \Sigma$ ➔Q Is total function ➔transition function
    - ✓ $q_o \in Q$: is Initial state
    - ✓ $F \subseteq Q$ is set of Final state
- A non-deterministic finite accepter or NFA is defined by the quintuple
  $M=(Q,\Sigma,\delta,q_o, F)$
    - ✓ Q is finite state

- ✓ Σ is input symbol
- ✓ δ :Q* ΣU{λ} ➔2^Q Is total function ➔transition function
- ✓ $q_o \in Q$: is Initial state
- ✓ $F \subseteq Q$ is set of Final state

- One way of describing regular languages is via the notation of regular expressions.
- NLP, as an area of computer science, has greatly benefitted from regexps: they are used in
  - ✓ Phonology,
  - ✓ Morphology,
  - ✓ Text Analysis,
  - ✓ Information Extraction
  - ✓ Speech Recognition.

Character	Regular-expression meaning
.	Any character, including whitespace or numeric
?	Zero or one of the preceding character
*	Zero or more of the preceding character
+	One or more of the preceding character
^	Negation or complement

- There are three operators used to build regular expressions:

❖ Union

- ❖ R|S – L(R|S) = L(R) ∪ L(S)

- ❖ Concatenation
    - ❖ RS – L(RS) = {rs, r ∈ R and s ∈ S}
- ❖ Closure
    - ❖ R* – L(R*) = {ε,R,RR,RRR,…}

- Equivalence of Regular Expressions and Finite Automata

- ❑ The languages accepted by finite automata are equivalent to those generated by regular expressions
- ❑ Given any regular expression R, there exists a finite state automata M such that L(M) = L(R).
- ❑ Given any finite state automata M, there exists a regular expression R such that L(R) = L(M)

Give Applications of Finite Automata

String Processing

Consider finding all occurrences of a short string (pattern string) within a long string (text string).

This can be done by processing the text through a DFA: the DFA for all strings that end with the pattern string. Each time the accept state is reached, the current position in the text is output.

Finite-State Machines

A finite-state machine is an FA together with actions on the arcs.

Statecharts

Statecharts model tasks as a set of states and actions. They extend FA diagrams.

Lexical Analysis

In compiling a program, the first step is lexical analysis. This **isolates keywords, identi**fiers etc., while
eliminating irrelevant symbols. A token is a category, for example “identifier”, “relation operator” or **speci**fic keyword

Derivation Tree

Derivation trees (also called "parse trees" in Sethi's book) are a way to represent the generation of strings in a grammar. They also give information about the structure of the strings, i.e. the way they are organized in syntactical categories.

5. Convert the following regular expression into finite state automata (FSA/M)?

A. ac|ad|cd*

Step 1 − Construct a Transition diagram for a given RE by using Non-deterministic finite automata (NFA) with ε moves.

Step 2 − Convert NFA with ε to NFA without ε.

Step 3 − Convert the NFA to the equivalent Deterministic Finite Automata (DFA)

M=L(r)

L(r)=L(ac+ad+cd*)

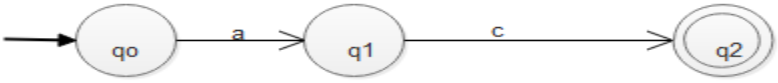
M=L(r1+r2+r3)

L(r1)=L(ac)……. Is {ac}

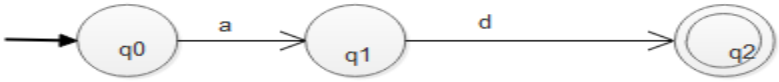
L(r2)=L(ad)……. Is {ad}

L(r3)=L(cd*)……. Is { c,d ,cd,cdd,cddd,cdddd,……}

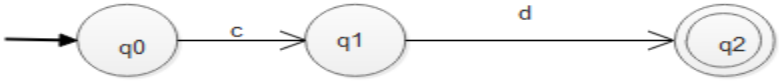
M=L(r1)U(l(r2)UL(r3)) ={ ad,ac,c,d ,cd,cdd,cddd,cdddd,……}



A



B

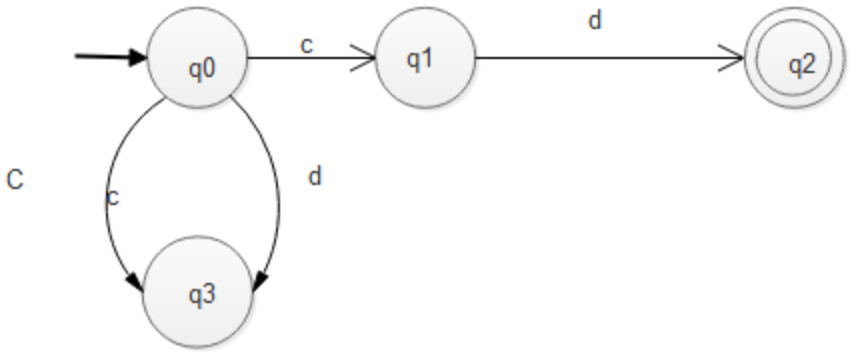


C

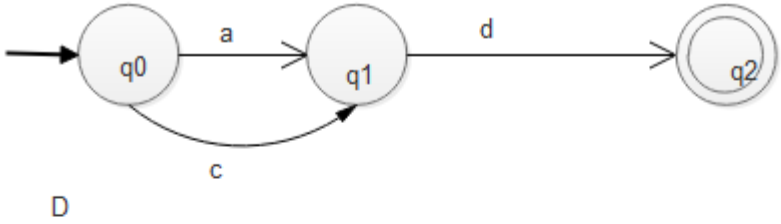
A string w is accepted by an NFA M if and only if there exists a path starting at q0 which is labeled by w and ends in a final state.

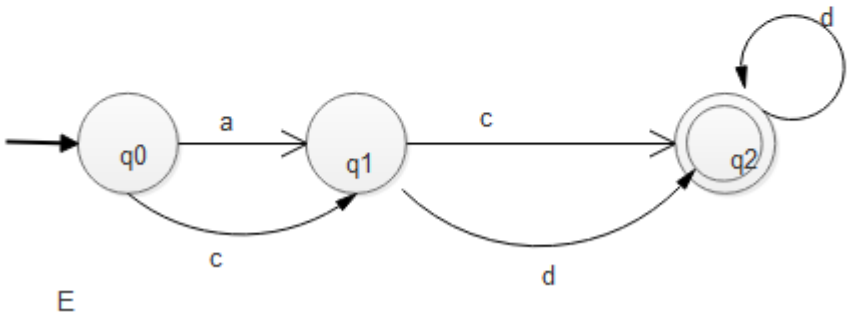
The language accepted by an NFA M is the set of all strings which are accepted by M and is denoted by L (M).
L(M)={w: δ(q0,w) F ≠ Φ}

So ,for the language generated by regular expression {c,d} is not accepted by NFS because the state at which c,d, will take us is not a final state.



So ,for the language generated by regular expression {c,d} is not accepted by NFS because the state at which c,d, will take us is not a final state.





6.Write a paragraph in any topic with only 100 words. Select five words from the paragraph and calculate probability of each word using

a. Chain rule
b. Markov assumption
c. Discuss what difference you have realized?
d. Discuss limitation of both Markov assumption and Chain rule?

Face Recognition with OpenCV and Python. What is face recognition Or what is recognition When you look at an apple fruit, your mind immediately tells you that this is an apple fruit. This process, your mind telling you that this is an apple fruit is recognition in simple words. So that is why Face Recognition with OpenCV and Python is good ,using human mind concept that react with environment. Interestingly when you look at your friend or a picture of him you look at his face first before looking at anything else. Face Recognition with OpenCV and Python do that.

b. chain rule

P (W total) =100

P (w1, w2, w3, w4, w5, w6)

=P (w1)*P (w2|w1)*P (w3|w1, w2)

*P(w4/w1,w2,w3)*P(w5/w1,w2,w3,w4)*P(w6/w1,w2,w3,w4,w5)

P (Face ,Recognition ,with, OpenCV ,and ,Python)

=P (5/100)*P (4/5)*P (4/4)*P (3/3)*P (3/3)*P (3/3)

=0.04

c. Markov assumption

P (wn | w1, w2, …, wn-1) ≈ P(wn | wn-1, wn-1)

P (w1, w2, w3, w4, w5, w6)

=P (w6/w5)*P (w6/w4, w5)*P (w6/w3, w4, w5)

*P (w6/w2, w3, w4, w5)*P (w6/w1, w2, w3, w4, w5)

P (Face ,Recognition ,with, OpenCV ,and ,Python)=

=P (Python |and)* P (Python| OpenCV ,and ,)* P (Python| with ,OpenCV ,and ,)* P (Python| Recognition ,with ,OpenCV ,and ,)* P (Python| Face ,Recognition with ,OpenCV ,and)

=(3/3)*(3/3)*(3/3)*(4/4)*(4/5)*(5/100)

=0.04

But **Markov assumption is said that there is no need to look to far history for getting** P (Face ,Recognition ,with, OpenCV ,and ,Python)=

=P (Python |and)* P (Python| OpenCV ,and ,)* P (Python| with ,OpenCV ,and)

=(3/3)*(3/3)*(3/3)

=1

d. Discuss what difference you have realized?

- Starting from the nearest neighbor words can increase the probability of approximation to the given word and reduce computational complexity.
- Markov predict the probability of item from short history
- Markov's rule have high probability than chain rule to search occurrence of next word

e. Discuss limitation of both Markov assumption and Chain rule?

There limitation of Chain rule

- It is possible to P(w/h),but it doesn't reduce computational complexity
- Finding the probability for a long sentence may not yield good outcome as the context may never occur in the corpus
- Use innovative ways to string words to form new sentences
- It's boring to search for too much words

There limitation of Markov rule

- There could be some error in the computation which could be minimal, smaller so on
- There is always possibility that the sentence we are forming using markov assumption need not be right every time