Module C: Session 2

fix - "p"

AR(p) ≡ $x_t = b + \sum_{i=1}^{p} w_i x_{t-i}$

AR(2)

$x_3 = b + w_1 x_2 + w_2 x_1$

$x_t(t=3) = b + \sum_{i=1}^{2} w_i x_{t_{3-i}}$ → $\underline{b + w^T x}$

p = 1

AR(1) :

$x_t = b + w_1 x_{t-1}$

$x_2 = b + w_1 x_1$
$x_3 = b + w_1 x_2$] → Next state depends only on the previous state.

→ idea similar to Markov's

→ What is range of y?

Regression → infinite options

Markov's Matrix → 7 options
↳ Roam around only a few number of states

Markov :-

O/P : Next state

How many next states? finite

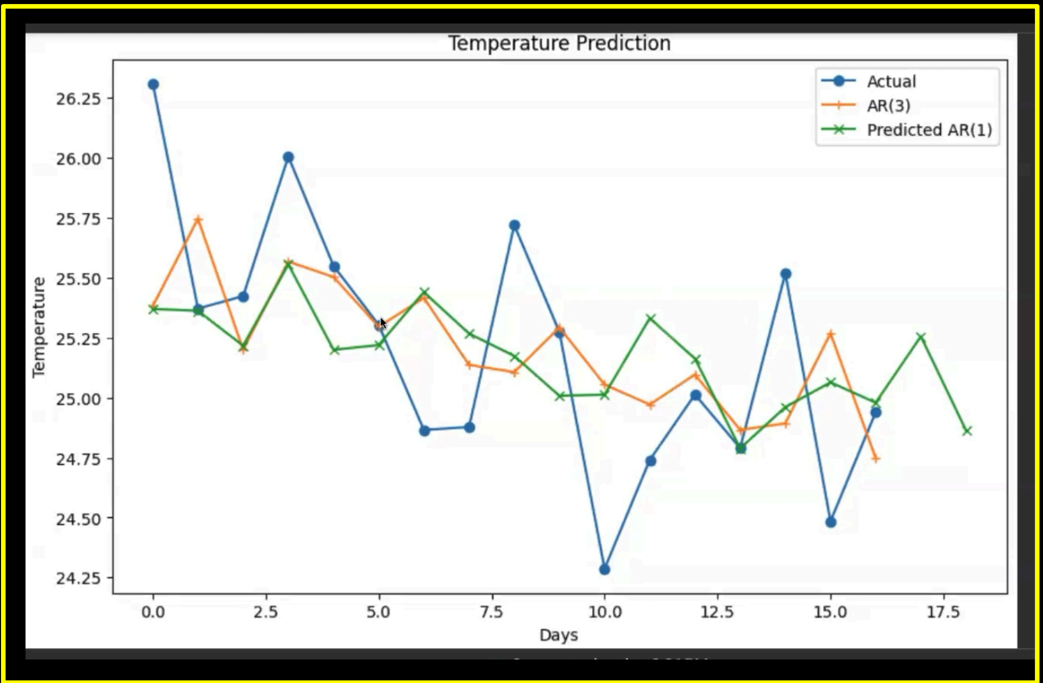
AUTO
REGRESSION

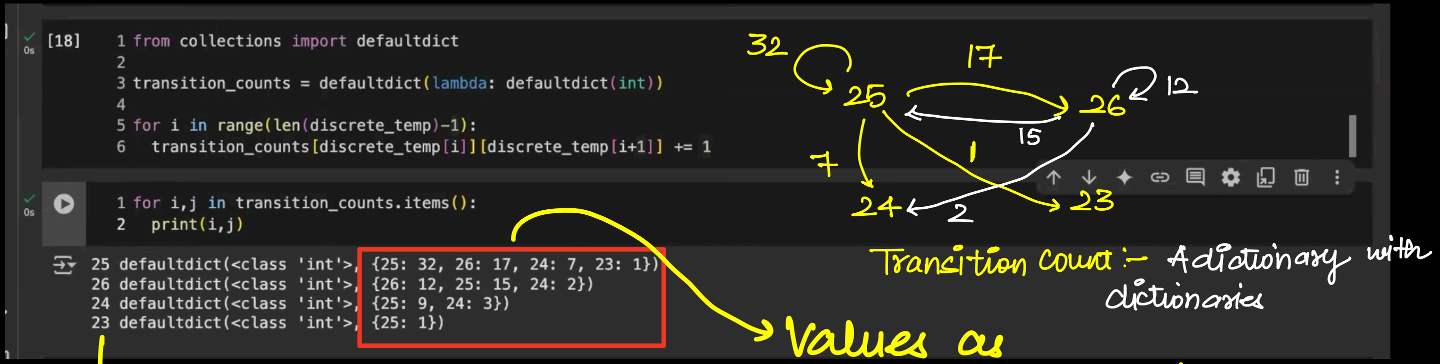
Sequences

Ar(p)
kind of
regression

Markov

finite
states
& need to get the
transition Matrix
↓
fix the size and get
the probability





```
from collections import defaultdict

transition_counts = defaultdict(lambda: defaultdict(int))

for i in range(len(discrete_temp)-1):
  transition_counts[discrete_temp[i]][discrete_temp[i+1]] += 1
```

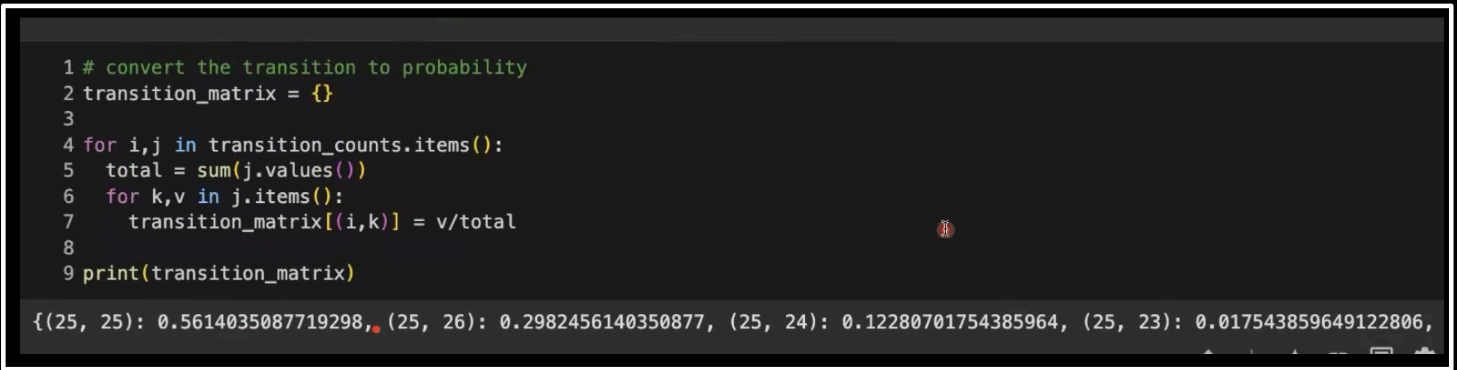
```
for i,j in transition_counts.items():
  print(i,j)
```

```
25 defaultdict(<class 'int'>, {25: 32, 26: 17, 24: 7, 23: 1})
26 defaultdict(<class 'int'>, {26: 12, 25: 15, 24: 2})
24 defaultdict(<class 'int'>, {25: 9, 24: 3})
23 defaultdict(<class 'int'>, {25: 1})
```

Transition count :- A dictionary with dictionaries

Values as dictionararies

Keys



```
# convert the transition to probability
transition_matrix = {}

for i,j in transition_counts.items():
  total = sum(j.values())
  for k,v in j.items():
    transition_matrix[(i,k)] = v/total

print(transition_matrix)
```

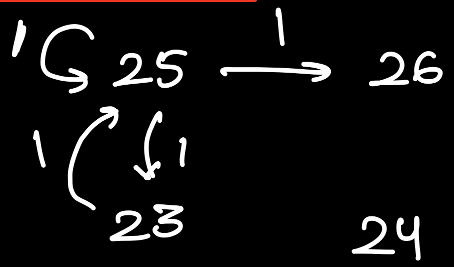
```
{(25, 25): 0.5614035087719298, (25, 26): 0.2982456140350877, (25, 24): 0.12280701754385964, (25, 23): 0.017543859649122806,
```

↳ constructing Markov's Matrix

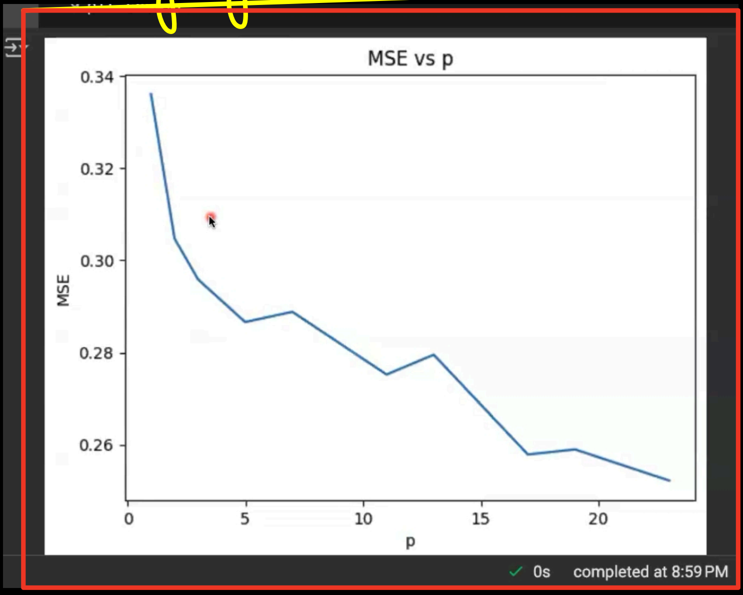
0	1	2	3	4	5	6
25	26	24	25	25	23	25

23 : 25 : 1
25 : 26 : 1 , 25 : 1, 23 : 1
26 : 24 : 1
24 : 25 : 1

$$\begin{bmatrix} 0.3 \\ 0.4 \\ 0.2 \\ 0.1 \end{bmatrix}$$



Changing values of p and change in error



✓ 0s completed at 8:59 PM

→ Consistency is more important

NLP :- NATURAL LANGUAGE PROCESSING

Text as one more input (other than numbers)

Computer can only understand numbers.

Somehow, convert text into number

Text to Numbers

(821)

I am enjoying my studies.

0 1 2 3 4

I love my country (242)

0 1 2 3

A Method to attach Numbers

English Dictionary

1 –	word
2 –	word
3 –	word
⋮	
242	country
821	enjoying

Coffee is great.

Car battery is dead, oh great. → NEXT LEVEL PROBLEM

Are both 'great' attached to same number?

Each word is assigned some number based on its index in a constructed dictionary.

e.g. I am from Mysore.
My name is Raghava.
I like eating Dosa.

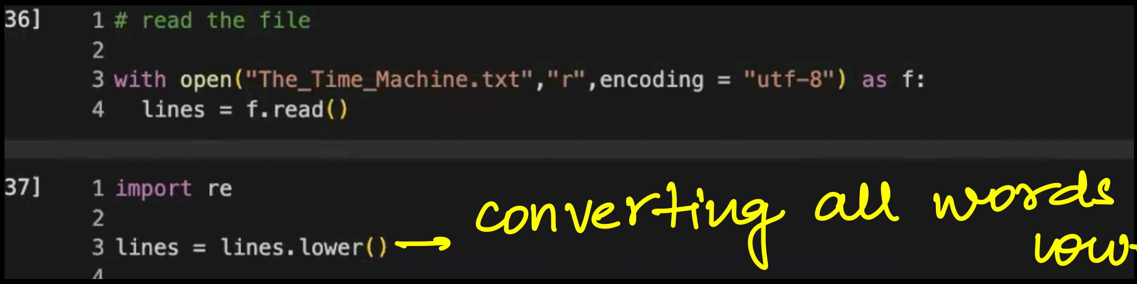
UNIQUE WORDS:-

I^{1} am^{2} from^{3} Mysore^{4} My^{5} name^{6} is^{7}
Raghava^{8} like^{9} eating^{10} Dosa^{11}

↳ Each word is assigned a number.

'TOKENIZATION'

All words together → VOCABLURY



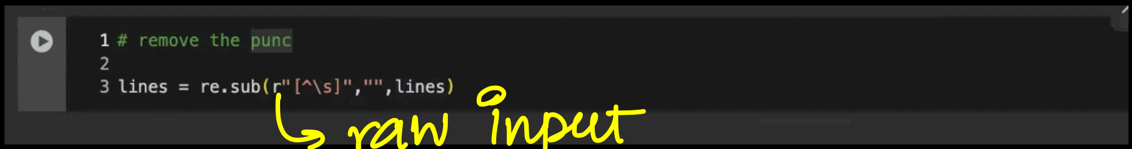
```
36]
# read the file

with open("The_Time_Machine.txt","r",encoding = "utf-8") as f:
  lines = f.read()
```

```
37]
import re

lines = lines.lower()
```

→ converting all words to lower case



```
# remove the punc

lines = re.sub(r"[^\s]","",lines)
```

↳ raw input