LAMPIRAN

Lampiran 1 Data Inflasi di Kabupaten Buleleng tahun 2019-2024

t	Bulan	Aktual	t	Bulan	Aktual
1	Januari 2019	0.58	33	September 2021	-0.45
2	Februari 2019	-0.34	34	Oktober 2021	0.08
3	Maret 2019	0.35	35	Nopember 2021	0.12
4	April 2019	0.39	36	Desember 2021	1.7
5	Mei 2019	0.28	37	Januari 2022	0.63
6	Juni 2019	0.02	38	Februari 2022	-0.84
7	Juli 2019	1.03	39	Maret 2022	1.27
8	Agustus 2019	0.62	40	April 2022	0.89
9	September 2019	-0.87	41	Mei 2022	0.58
10	Oktober 2019	-0.14	42	Juni 2022	2.2
11	Nopember 2019	0.22	43	Juli 2022	0.48
12	Desember 2019	0.27	44	Agustus 2022	-1.48
13	Januari 2020	0.67	45	September 2022	0.35
14	Februari 2020	0.7	46	Oktober 2022	-0.16
15	Maret 2020	0.15	47	Nopember2022	0.07
16	April 2020	-0.36	48	Desember 2022	0.59
17	Mei 2020	-0.22	49	Januari 2023	0.59
18	Juni 2020	0.32	50	Februari 2023	0.29
19	Juli 2020	0.11	51	Maret 2023	0.42
20	Agustus 2020	-0.42	52	April 2023	-0.22
21	September 2020	0.27	53	Mei 2023	0.37
22	Oktober 2020	-0.21	54	Juni 2023	0.22
23	Nopember 2020	0.37	55	Juli2023	0.25
24	Desember 2020	1.08	56	Agustus 2023	0.27
25	Januari 2021	0.94	57	September2023	-0.05
26	Februari 2021	0.22	58	Oktober 2023	0.44
27	Maret 2021	0.81	59	Nopember 2023	0.87
28	April 2021	-0.15	60	Desember 2023	0.43
29	Mei 2021	-0.5	61	Januari 2024	-0.22
30	Juni 2021	-0.52	62	Februari 2024	0.51
31	Juli 2021	0.19	63	Maret 2024	0.89
32	Agustus 2021	-0.07			

Lampiran 2 Hasil Fuzzyfikasi Data Inflasi

t	Aktual	*Fuzzy*	t	Aktual	*Fuzzy*
1	0.58	A4	33	-0.45	A3
2	-0.34	A3	34	0.08	A4
3	0.35	A4	35	0.12	A4
4	0.39	A4	36	1.7	A6
5	0.28	A4	37	0.63	A5
6	0.02	A4	38	-0.84	A2
7	1.03	A5	39	1.27	A6
8	0.62	A5	40	0.89	A5
9	-0.87	A2	41	0.58	A4
10	-0.14	A3	42	2.2	A7
11	0.22	A4	43	0.48	A4
12	0.27	A4	44	-1.48	A1
13	0.67	A5	45	0.35	A4
14	0.7	A5	46	-0.16	A3
15	0.15	A4	47	0.07	A4
16	-0.36	A3	48	0.59	A4
17	-0.22	A3	49	0.59	A4
18	0.32	A4	50	0.29	A4
19	0.11	A4	51	0.42	A4
20	-0.42	A3	52	-0.22	A3
21	0.27	A4	53	0.37	A4
22	-0.21	A3	54	0.22	A4
23	0.37	A4	55	0.25	A4
24	1.08	A5	56	0.27	A4
25	0.94	A5	57	-0.05	A3
26	0.22	A4	58	0.44	A4
27	0.81	A5	59	0.87	A5
28	-0.15	A3	60	0.43	A4
29	-0.5	A3	61	-0.22	A3
30	-0.52	A3	62	0.51	A4
31	0.19	A4	63	0.89	A5
32	-0.07	A3			

Lampiran 3 Hasil *Fuzzy Logical Relationsip* (FLR) dari Data Inflasi

t	Aktual	FLR			t	Aktual	*FLR*		
1	0.58	A4	→	A3	33	-0.45	A3	→	A4
2	-0.34	A3	→	A4	34	0.08	A4	→	A4
3	0.35	A4	→	A4	35	0.12	A4	→	A6
4	0.39	A4	→	A4	36	1.7	A6	→	A5
5	0.28	A4	→	A4	37	0.63	A5	→	A2
6	0.02	A4	→	A5	38	-0.84	A2	→	A6
7	1.03	A5	→	A5	39	1.27	A6	→	A5
8	0.62	A5	→	A2	40	0.89	A5	→	A4
9	-0.87	A2	→	A3	41	0.58	A4	→	A7
10	-0.14	A3	→	A4	42	2.2	A7	→	A4
11	0.22	A4	→	A4	43	0.48	A4	→	A1
12	0.27	A4	→	A5	44	-1.48	A1	→	A4
13	0.67	A5	→	A5	45	0.35	A4	→	A3
14	0.7	A5	→	A4	46	-0.16	A3	→	A4
15	0.15	A4	→	A3	47	0.07	A4	→	A4
16	-0.36	A3	→	A3	48	0.59	A4	→	A4
17	-0.22	A3	→	A4	49	0.59	A4	→	A4
18	0.32	A4	→	A4	50	0.29	A4	→	A4
19	0.11	A4	→	A3	51	0.42	A4	→	A3
20	-0.42	A3	→	A4	52	-0.22	A3	→	A4
21	0.27	A4	→	A3	53	0.37	A4	→	A4
22	-0.21	A3	→	A4	54	0.22	A4	→	A4
23	0.37	A4	→	A5	55	0.25	A4	→	A4
24	1.08	A5	→	A5	56	0.27	A4	→	A3
25	0.94	A5	→	A4	57	-0.05	A3	→	A4
26	0.22	A4	→	A5	58	0.44	A4	→	A5
27	0.81	A5	→	A3	59	0.87	A5	→	A4
28	-0.15	A3	→	A3	60	0.43	A4	→	A3
29	-0.5	A3	→	A3	61	-0.22	A3	→	A4
30	-0.52	A3	→	A4	62	0.51	A4	→	A5
31	0.19	A4	→	A3	63	0.89	A5		
32	-0.07	A3	→	A3					

Lampiran 4 Hasil Prediksi Awal $F(t)$ *Markov Chain* dari Data Inflasi

t	Aktual	$F(t)$	t	Aktual	$F(t)$
1	0.58	*	33	-0.45	0,15076
2	-0.34	0,29074	34	0.08	0,15076
3	0.35	0,15076	35	0.12	0,29074
4	0.39	0,29074	36	1.7	0,29074
5	0.28	0,29074	37	0.63	0,90714
6	0.02	0,29074	38	-0.84	0,19057
7	1.03	0,29074	39	1.27	0,60857
8	0.62	0,19057	40	0.89	0,90714
9	-0.87	0,19057	41	0.58	0,19057
10	-0.14	0,60857	42	2.2	0,29074
11	0.22	0,15076	43	0.48	0,31
12	0.27	0,29074	44	-1.48	0,29074
13	0.67	0,29074	45	0.35	0,31
14	0.7	0,19057	46	-0.16	0,29074
15	0.15	0,19057	47	0.07	0,15076
16	-0.36	0,29074	48	0.59	0,29074
17	-0.22	0,15076	49	0.59	0,29074
18	0.32	0,15076	50	0.29	0,29074
19	0.11	0,29074	51	0.42	0,29074
20	-0.42	0,29074	52	-0.22	0,29074
21	0.27	0,15076	53	0.37	0,15076
22	-0.21	0,29074	54	0.22	0,29074
23	0.37	0,15076	55	0.25	0,29074
24	1.08	0,29074	56	0.27	0,29074
25	0.94	0,19057	57	-0.05	0,29074
26	0.22	0,19057	58	0.44	0,15076
27	0.81	0,29074	59	0.87	0,29074
28	-0.15	0,19057	60	0.43	0,19057
29	-0.5	0,15076	61	-0.22	0,29074
30	-0.52	0,15076	62	0.51	0,15076
31	0.19	0,15076	63	0.89	0,29074
32	-0.07	0,29074			

Lampiran 5 Hasil Perhitungan D_t *Markov Chain* dari Data Inflasi

t	Aktual	$F(t)$	D_{t1}	D_{t1}	t	Aktual	$F(t)$	D_{t1}	D_{t1}
1	0.58	*	*	*	33	-0.45	0,15076	0	0
2	-0.34	0,29074	-0,2986	-0,2986	34	0.08	0,15076	0,29857	0,29857
3	0.35	0,15076	0,29857	0,29857	35	0.12	0,29074	0	0
4	0.39	0,29074	0	0	36	1.7	0,29074	0,29857	0,59714
5	0.28	0,29074	0	0	37	0.63	0,90714	-0,2986	-0,2986
6	0.02	0,29074	0	0	38	-0.84	0,19057	-0,2986	-0,8957
7	1.03	0,29074	0,29857	0,29857	39	1.27	0,60857	0,29857	1,19429
8	0.62	0,19057	0	0	40	0.89	0,90714	-0,2986	-0,2986
9	-0.87	0,19057	-0,2986	-0,8957	41	0.58	0,19057	-0,2986	-0,2986
10	-0.14	0,60857	0,29857	0,29857	42	2.2	0,29074	0,29857	0,89571
11	0.22	0,15076	0,29857	0,29857	43	0.48	0,31	-0,2986	-0,8957
12	0.27	0,29074	0	0	44	-1.48	0,29074	-0,2986	-0,8957
13	0.67	0,29074	0,29857	0,29857	45	0.35	0,31	0,29857	0,89571
14	0.7	0,19057	0	0	46	-0.16	0,29074	-0,2986	-0,2986
15	0.15	0,19057	-0,2986	-0,2986	47	0.07	0,15076	0,29857	0,29857
16	-0.36	0,29074	-0,2986	-0,2986	48	0.59	0,29074	0	0
17	-0.22	0,15076	0	0	49	0.59	0,29074	0	0
18	0.32	0,15076	0,29857	0,29857	50	0.29	0,29074	0	0
19	0.11	0,29074	0	0	51	0.42	0,29074	0	0
20	-0.42	0,29074	-0,2986	-0,2986	52	-0.22	0,29074	-0,2986	-0,2986
21	0.27	0,15076	0,29857	0,29857	53	0.37	0,15076	0,29857	0,29857
22	-0.21	0,29074	-0,2986	-0,2986	54	0.22	0,29074	0	0
23	0.37	0,15076	0,29857	0,29857	55	0.25	0,29074	0	0
24	1.08	0,29074	0,29857	0,29857	56	0.27	0,29074	0	0
25	0.94	0,19057	0	0	57	-0.05	0,29074	-0,2986	-0,2986
26	0.22	0,19057	-0,2986	-0,2986	58	0.44	0,15076	0,29857	0,29857
27	0.81	0,29074	0,29857	0,29857	59	0.87	0,29074	0,29857	0,29857
28	-0.15	0,19057	-0,2986	-0,5971	60	0.43	0,19057	-0,2986	-0,2986
29	-0.5	0,15076	0	0	61	-0.22	0,29074	-0,2986	-0,2986
30	-0.52	0,15076	0	0	62	0.51	0,15076	0,29857	0,29857
31	0.19	0,15076	0,29857	0,29857	63	0.89	0,29074	0,29857	0,29857
32	-0.07	0,29074	-0,2986	-0,2986					

Lampiran 6 Hasil Akhir Prediksi $F'(t)$ dari *Fuzzy Time Series Markov Chain* pada Data Inflasi

t	Aktual	$F(t)$	$F'(t)$	t	Aktual	$F(t)$	$F'(t)$
1	0.58	*	*	33	-0.45	0,15076	0,15076
2	-0.34	0,29074	-0,3064	34	0.08	0,15076	0,7479
3	0.35	0,15076	0,7479	35	0.12	0,29074	0,29074
4	0.39	0,29074	0,29074	36	1.7	0,29074	1,18645
5	0.28	0,29074	0,29074	37	0.63	0,90714	0,31
6	0.02	0,29074	0,29074	38	-0.84	0,19057	-1,0037
7	1.03	0,29074	0,88788	39	1.27	0,60857	2,10143
8	0.62	0,19057	0,19057	40	0.89	0,90714	0,31
9	-0.87	0,19057	-1,0037	41	0.58	0,19057	-0,4066
10	-0.14	0,60857	1,20571	42	2.2	0,29074	1,48502
11	0.22	0,15076	0,7479	43	0.48	0,31	-0,8843
12	0.27	0,29074	0,29074	44	-1.48	0,29074	-0,9035
13	0.67	0,29074	0,88788	45	0.35	0,31	1,50429
14	0.7	0,19057	0,19057	46	-0.16	0,29074	-0,3064
15	0.15	0,19057	-0,4066	47	0.07	0,15076	0,7479
16	-0.36	0,29074	-0,3064	48	0.59	0,29074	0,29074
17	-0.22	0,15076	0,15076	49	0.59	0,29074	0,29074
18	0.32	0,15076	0,7479	50	0.29	0,29074	0,29074
19	0.11	0,29074	0,29074	51	0.42	0,29074	0,29074
20	-0.42	0,29074	-0,3064	52	-0.22	0,29074	-0,3064
21	0.27	0,15076	0,7479	53	0.37	0,15076	0,7479
22	-0.21	0,29074	-0,3064	54	0.22	0,29074	0,29074
23	0.37	0,15076	0,7479	55	0.25	0,29074	0,29074
24	1.08	0,29074	0,88788	56	0.27	0,29074	0,29074
25	0.94	0,19057	0,19057	57	-0.05	0,29074	-0,3064
26	0.22	0,19057	-0,4066	58	0.44	0,15076	0,7479
27	0.81	0,29074	0,88788	59	0.87	0,29074	0,88788
28	-0.15	0,19057	-0,7051	60	0.43	0,19057	-0,4066
29	-0.5	0,15076	0,15076	61	-0.22	0,29074	-0,3064
30	-0.52	0,15076	0,15076	62	0.51	0,15076	0,7479
31	0.19	0,15076	0,7479	63	0.89	0,29074	0,88788
32	-0.07	0,29074	-0,3064				

Lampiran 7 Hasil Perhitungan dari $D_z, X_i, XX_i, Y_i, dan\ YY_i$ pada Data Inflasi

t	Bulan	Aktual	D_z	X_i	XX_i	Y_i	YY_i
1	Januari 2019	0.58	*	*	*	*	*
2	Februari 2019	-0.34	*	*	*	*	*
3	Maret 2019	0.35	0,23	0,58	0,12	0,465	0,235
4	April 2019	0.39	0,65	1,04	-0,26	0,715	0,065
5	Mei 2019	0.28	0,07	0,35	0,21	0,315	0,245
6	Juni 2019	0.02	0,15	0,17	-0,13	0,095	-0,055
7	Juli 2019	1.03	0,75	1,78	0,28	1,405	0,655
8	Agustus 2019	0.62	0,6	1,22	0,02	0,92	0,32
9	September 2019	-0.87	1,08	0,21	-1,95	-0,33	-1,41
10	Oktober 2019	-0.14	0,76	0,62	-0,9	0,24	-0,52
11	Nopember 2019	0.22	0,37	0,59	-0,15	0,405	0,035
12	Desember 2019	0.27	0,31	0,58	-0,04	0,425	0,115
13	Januari 2020	0.67	0,35	1,02	0,32	0,845	0,495
14	Februari 2020	0.7	0,37	1,07	0,33	0,885	0,515
15	Maret 2020	0.15	0,52	0,67	-0,37	0,41	-0,11
16	April 2020	-0.36	0,04	-0,32	-0,4	-0,34	-0,38
17	Mei 2020	-0.22	0,37	0,15	-0,59	-0,035	-0,405
18	Juni 2020	0.32	0,4	0,72	-0,08	0,52	0,12
19	Juli 2020	0.11	0,33	0,44	-0,22	0,275	-0,055
20	Agustus 2020	-0.42	0,32	-0,1	-0,74	-0,26	-0,58
21	September 2020	0.27	0,16	0,43	0,11	0,35	0,19
22	Oktober 2020	-0.21	0,21	-3E-17	-0,42	-0,105	-0,315
23	Nopember 2020	0.37	0,1	0,47	0,27	0,42	0,32
24	Desember 2020	1.08	0,13	1,21	0,95	1,145	1,015
25	Januari 2021	0.94	0,57	1,51	0,37	1,225	0,655
26	Februari 2021	0.22	0,58	0,8	-0,36	0,51	-0,07
27	Maret 2021	0.81	0,13	0,94	0,68	0,875	0,745
28	April 2021	-0.15	0,37	0,22	-0,52	0,035	-0,335
29	Mei 2021	-0.5	0,61	0,11	-1,11	-0,195	-0,805
30	Juni 2021	-0.52	0,33	-0,19	-0,85	-0,355	-0,685
31	Juli 2021	0.19	0,69	0,88	-0,5	0,535	-0,155
32	Agustus 2021	-0.07	0,45	0,38	-0,52	0,155	-0,295
33	September 2021	-0.45	0,12	-0,33	-0,57	-0,39	-0,51
34	Oktober 2021	0.08	0,15	0,23	-0,07	0,155	0,005
35	Nopember 2021	0.12	0,49	0,61	-0,37	0,365	-0,125

Lampiran 8 (Lanjutan)

t	Bulan	Aktual	D_z	X_i	XX_i	Y_i	YY_i
36	Desember 2021	1.7	1,54	3,24	0,16	2,47	0,93
37	Januari 2022	0.63	0,51	1,14	0,12	0,885	0,375
38	Februari 2022	-0.84	0,4	-0,44	-1,24	-0,64	-1,04
39	Maret 2022	1.27	0,64	1,91	0,63	1,59	0,95
40	April 2022	0.89	1,73	2,62	-0,84	1,755	0,025
41	Mei 2022	0.58	0,07	0,65	0,51	0,615	0,545
42	Juni 2022	2.2	1,31	3,51	0,89	2,855	1,545
43	Juli 2022	0.48	0,1	0,58	0,38	0,53	0,43
44	Agustus 2022	-1.48	0,24	-1,24	-1,72	-1,36	-1,6
45	September 2022	0.35	0,13	0,48	0,22	0,415	0,285
46	Oktober 2022	-0.16	1,32	1,16	-1,48	0,5	-0,82
47	Nopember2022	0.07	0,28	0,35	-0,21	0,21	-0,07
48	Desember 2022	0.59	0,29	0,88	0,3	0,735	0,445
49	Januari 2023	0.59	0,52	1,11	0,07	0,85	0,33
50	Februari 2023	0.29	0,3	0,59	-0,01	0,44	0,14
51	Maret 2023	0,42	0,17	0,59	0,25	0,505	0,335
52	April 2023	-0,22	0,51	0,29	-0,73	0,035	-0,475
53	Mei 2023	0,37	0,05	0,42	0,32	0,395	0,345
54	Juni 2023	0,22	0,44	0,66	-0,22	0,44	2,8E-17
55	Juli2023	0,25	0,12	0,37	0,13	0,31	0,19
56	Agustus 2023	0,27	0,01	0,28	0,26	0,275	0,265
57	September2023	-0,05	0,3	0,25	-0,35	0,1	-0,2
58	Oktober 2023	0,44	0,17	0,61	0,27	0,525	0,355
59	Nopember 2023	0,87	0,06	0,93	0,81	0,9	0,84
60	Desember 2023	0,43	0,01	0,44	0,42	0,435	0,425
61	Januari 2024	-0,22	0,21	-0,01	-0,43	-0,115	-0,325
62	Februari 2024	0,51	0,08	0,59	0,43	0,55	0,47
63	Maret 2024	0,89	0,35	1,24	0,54	1,065	0,715

Lampiran 2 Hasil Prediksi *Fuzzy Time Series S. R. Singh* pada Data Inflasi

t	Aktual	$F(t)$	t	Aktual	$F(t)$
1	0.58	*	33	-0.45	-0,4364
2	-0.34	*	34	0.08	-0,4364
3	0.35	0,16071	35	0.12	-0,4364
4	0.39	0,16071	36	1.7	0,75786
5	0.28	0,16071	37	0.63	0,16071
6	0.02	-0,4364	38	-0.84	-1,0336
7	1.03	0,75786	39	1.27	0,75786
8	0.62	0,16071	40	0.89	0,16071
9	-0.87	-1,6307	41	0.58	0,16071
10	-0.14	-0,4364	42	2.2	1,355
11	0.22	0,16071	43	0.48	0,16071
12	0.27	0,16071	44	-1.48	-1,6307
13	0.67	0,16071	45	0.35	0,16071
14	0.7	0,16071	46	-0.16	-1,0336
15	0.15	-0,4364	47	0.07	-0,4364
16	-0.36	-0,4364	48	0.59	0,16071
17	-0.22	-0,4364	49	0.59	0,16071
18	0.32	0,16071	50	0.29	0,16071
19	0.11	-0,4364	51	0.42	0,16071
20	-0.42	-0,4364	52	-0.22	-0,4364
21	0.27	0,16071	53	0.37	0,16071
22	-0.21	-0,4364	54	0.22	-0,4364
23	0.37	0,16071	55	0.25	0,16071
24	1.08	0,75786	56	0.27	0,16071
25	0.94	0,75786	57	-0.05	-0,4364
26	0.22	-0,4364	58	0.44	0,16071
27	0.81	0,75786	59	0.87	0,75786
28	-0.15	-0,4364	60	0.43	0,16071
29	-0.5	-1,0336	61	-0.22	-0,4364
30	-0.52	-1,0336	62	0.51	0,16071
31	0.19	-0,4364	63	0.89	0,75786
32	-0.07	-0,4364			

Lampiran 3 *Coding* Pemrograman *Phyton* untuk Pengolahan Data

```
# Import library yang diperlukan
        import pandas as pd
        import numpy as np
        import plotly.graph_objects as go
        from plotly.subplots import make_subplots
        from sklearn.metrics import mean_absolute_error, mean_squared_error
# Membaca file Excel
        df = pd.read_excel('datainflasi.xlsx')
# Langkah 1 menentukan himpunan semesta U
        # Menghitung D_min and D_max
                D_min = df['aktual'].min()
                D_max = df['aktual'].max()
        # Menentukan D_1 and D_2
                D_1 = 0.3  # Example value, adjust as needed
                D_2 = 0.2  # Example value, adjust as needed
        # Menampilkan D_min, D_max, D_1, D2
                print("D_min:", D_min)
                print("D_max:", D_max)
                print("D_1:", D_1)
                print("D_2:", D_2)
        # Menghitung himpunan semesta U
                U = [D_min - D_1, D_max + D_2]
        print("Universe U:", U)
```

Lampiran 23 (Lanjutan)

```
# Langkah 2 Membagi himpunan semesta U menjadi n interval yang sama menggunakan aturan Sturges
    # Menghitung jumlah data historis (N)
        N = len(df)
    # Menghitung jumlah interval (n) menggunakan aturan Sturges
        n = int(np.ceil(1 + 3.322 * np.log10(N)))
        print('interval :', n)
    # Menghitung panjang interval (l)
        l = (U[1] - U[0]) / n
        print('panjang interval :',l)
    # Membuat list untuk menyimpan interval
        intervals = []
        for i in range(n):
         interval_start = U[0] + i * l
         interval_end = U[0] + (i + 1) * l
         intervals.append([interval_start, interval_end])
    # Menampilkan interval-interval
        for i, interval in enumerate(intervals):
         print(f"u_{i+1} = {interval}")
    # Menghitung nilai tengah setiap interval
        midpoints = []
        for interval in intervals:
         midpoint = (interval[0] + interval[1]) / 2
         midpoints.append(midpoint)
    # Menampilkan nilai tengah
print("NILAI TENGAH")
```

Lampiran 23 (Lanjutan)

```
                for i, midpoint in enumerate(midpoints):
                  print(f"u_{i+1}: {midpoint}")
# LANGKAH 3
        # Menentukan himpunan-himpunan fuzzy
                membership_matrix = np.zeros((n, n))
                for i in range(n):
                  for j in range(n):
                    if i == j:
                      membership_matrix[i, j] = 1
                    elif j == i - 1 or j == i + 1:
                      membership_matrix[i, j] = 0.5
                    else:
                      membership_matrix[i, j] = 0
        # Menampilkan matriks derajat keanggotaan
                print("Membership Matrix:")
                print(membership_matrix)
        # Menampilkan himpunan-himpunan fuzzy
                fuzzy_sets = []
                for i in range(n):
                  fuzzy_set = []
                  for j in range(n):
                    if membership_matrix[i, j] >= 1:
                      fuzzy_set.append((f"A_{j+1}", membership_matrix[i, j]))
                  fuzzy_sets.append(fuzzy_set)
        # Menampilkan himpunan-himpunan fuzzy
for i, fuzzy_set in enumerate(fuzzy_sets):
```

Lampiran 23 (Lanjutan)

```
        print(f"Fuzzy Set A_{i+1}: {fuzzy_set}")
        # Fuzzifikasi data historis
                fuzzy_values = []
                for value in df['aktual']:
                  for i, interval in enumerate(intervals):
                    if interval[0] <= value < interval[1]:
                      fuzzy_values.append(f"A_{i+1}")
                      break
        # Menambahkan kolom fuzzy ke dallam DataFrame
                df['fuzzy'] = fuzzy_values
        # Menampilkan DataFrame dengan kolom fuzzy
                print(df)
# Langkah 4
        # Mencari Fuzzy Logical Relationships (FLR)
                flrs = []
                for i in range(1, len(df)):
                  current_fuzzy = df['fuzzy'][i]
                  prev_fuzzy = df['fuzzy'][i - 1]
                  flrs.append(f"{prev_fuzzy} -> {current_fuzzy}")
        # Menambahkan kolom FLR ke dallam DataFrame
                df['FLR'] = pd.Series(flrs)
        # Menampilkan DataFrame dengan kolom FLR
                print(df)
```

Lampiran 23 (Lanjutkan)

```
# Langkah 5
        # Mengelompokkan FLRG
                flrg_dict = {}
                for flr in flrs:
                  if ' -> ' in flr:  # Check if delimiter is present
                    lhs, rhs = flr.split(' -> ')
                    if lhs not in flrg_dict:
                      flrg_dict[lhs] = []
                    flrg_dict[lhs].append(rhs)
        # Menampilkan FLRG
                for lhs, rhs_list in flrg_dict.items():
                  print(f"{lhs} -> {', '.join(rhs_list)}")
```

Lampiran 4 Coding Pengolahan Data *Fuzzy Time Series Markov Chain*

```
# Langkah 6
    # Menghitung matriks probabilitas transisi R
        transition_counts = {}
        for lhs, rhs_list in flrg_dict.items():
          transition_counts[lhs] = {rhs: 0 for rhs in flrg_dict.keys()}
          for rhs in rhs_list:
            if rhs not in transition_counts[lhs]:
               transition_counts[lhs][rhs] = 0
            transition_counts[lhs][rhs] += 1
        transition_matrix = {}
        for lhs, counts in transition_counts.items():
          total_count = sum(counts.values())
          transition_matrix[lhs] = {rhs: count / total_count for rhs, count in
        counts.items()}
    # Menampilkan matriks probabilitas transisi R
        print("Transition Matrix:")
        for lhs, probabilities in transition_matrix.items():
          print(f"{lhs}: {probabilities}")
    #Prediksi Awal
        def calculate_forecast(t, transition_matrix, midpoints, df):
          if t == 0:
            return None  # Tidak ada prediksi untuk periode pertama kali
          else:
            prev_fuzzy = df['fuzzy'][t - 1]
            if prev_fuzzy in transition_matrix:
              if len(transition_matrix[prev_fuzzy]) == 1:  # transisi One-to-one
```

Lampiran 24 (Lanjutan)

```
predicted_fuzzy = next(iter(transition_matrix[prev_fuzzy]))
                predicted_value = midpoints[int(predicted_fuzzy.split('_')[1]) - 1]
                return predicted_value
                else:  # transisi One-to-many
                probabilities = transition_matrix[prev_fuzzy]
                weighted_sum = 0
                for i, (fuzzy_state, probability) in enumerate(probabilities.items()):
                        index = int(fuzzy_state.split('_')[1]) - 1
                        if i == index:  # Y(t-1) * P_ii term
                          weighted_sum += df['aktual'][t - 1] * probability
                        else:  # m_j * P_ij terms
                          weighted_sum += midpoints[index] * probability
                      return weighted_sum
                    else:
                        return None  # Untuk kasus dimana status sebelumnya tidak
                memiliki transisi
        # Hitung prediksi awal untuk semua periode waktu
                forecasts = []
                for t in range(1, len(df)):
                 forecast = calculate_forecast(t, transition_matrix, midpoints, df)
                 forecasts.append(forecast)
        # menambahkan data hasil prediksi awal ke dataframe
                df['prediksi_awal'] = pd.Series(forecasts, index=df.index[1:])
        # menampilkan dataframe dengan kolom prediksi awal
print(df)
```

Lampiran 24 (Lanjutan)

```
#Langkah 7
        def calculate_adjustment(t, df, l, n):
          if t == 0:
            return 0  # tidak ada penyesuaian untuk periode waktu pertama
          else:
            prev_fuzzy = df['fuzzy'][t - 1]
            current_fuzzy = df['fuzzy'][t]
            prev_index = int(prev_fuzzy.split('_')[1]) - 1
            current_index = int(current_fuzzy.split('_')[1]) - 1
            if prev_index > current_index:  # Transisi Ke bawah
                    if prev_fuzzy in transition_matrix and current_fuzzy in
        transition_matrix[prev_fuzzy]:
                return -l / 2  # D_t1
            elif prev_index < current_index:  # Transisi Ke Atas
                    if prev_fuzzy in transition_matrix and current_fuzzy in
        transition_matrix[prev_fuzzy]:
                return l / 2  # D_t1
            else:
              return 0  # Tidak ada penyesuaian untuk transisi lainnya
# Menghitung penyesuaian untuk semua periode waktu
        adjustments = []
        for t in range(1, len(df)):
          adjustment = calculate_adjustment(t, df, l, n)
          adjustments.append(adjustment)
# menambahkan kolom penyesuaian ke dataframe
        df['D_t1'] = pd.Series(adjustments, index=df.index[1:])
print(df)
```

Lampiran 24 (Lanjutan)

```
        def calculate_adjustment(t, df, l, n):
          if t == 0:
            return 0  # Tidak ada penyesuaian untuk peiode pertama
          else:
            prev_fuzzy = df['fuzzy'][t - 1]
            current_fuzzy = df['fuzzy'][t]
            prev_index = int(prev_fuzzy.split('_')[1]) - 1
            current_index = int(current_fuzzy.split('_')[1]) - 1
            if prev_index < current_index:  # Transisi ke atas
              if prev_index < n - 1:  # Transisi Maju dallam batas
                s = current_index - prev_index
                return (l / 2) * s # D_t2
              else:  # Transisi Lompatan mundur
                v = prev_index - current_index
                if 1 <= v <= prev_index:
                  return -(l / 2) * v  # D_t2
            else:
              return 0  # tidak ada penyesuaian untuk periode lain
        # Hitung penyesuaian untuk semua periode waktu setelah yang pertama
                adjustments2 = []
                for t in range(1, len(df)):
                  adjustments2 = calculate_adjustment(t, df, l, n)
                  adjustments.append(adjustment)
        # menambahka penyesuaian ke DataFrame
                df['D_t2'] = pd.Series(adjustments2, index=df.index[1:])
print(df)
```

Lampiran 24 (Lanjutan)

```
# Langkah 8 Markov Chain
# Menghitung prediksi yang disesuaikan F'(t)
        final_forecasts = []
        for t in range(1, len(df)):
          prev_fuzzy = df['fuzzy'][t - 1]
          current_fuzzy = df['fuzzy'][t]
          first_forecast = df['prediksi_awal'][t]
          adjusted_forecast = df['D_t1'][t] + df['D_t2'][t]
         if adjusted_forecast is not None:
          prev_index = int(prev_fuzzy.split('_')[1]) - 1
            current_index = int(current_fuzzy.split('_')[1]) - 1
   if prev_fuzzy in transition_matrix and len(transition_matrix[prev_fuzzy]) > 1:
    if current_fuzzy in transition_matrix[prev_fuzzy]:
     if prev_fuzzy in transition_matrix.get(current_fuzzy, {}):
      final_forecast = first_forecast + (adjusted_forecast)
   else:
    final_forecast = first_forecast
  final_forecasts.append(final_forecast)
# menambahkan hasil prediksi akhir ke dataframe
        df['prediksi_markov'] = pd.Series(final_forecasts, index=df.index[1:])
# menampilkan dataframe dengan kolom hasil prediksi akhir
print(df)
```

Lampiran 5 **Coding Pengolahan Data** ***Fuzzy Time Series S. R. Singh***

```
# Langkah 6
    # Menghitung D_z
        df['D_z'] = np.abs(np.abs(df['aktual'].diff()) -np.abs(df['aktual'].shift(1).diff()))
            # Menghitung X_i, XX_i, Y_i, YY_i
                    df['X_i'] = df['aktual'] + df['D_z']
                    df['XX_i'] = df['aktual'] - df['D_z']
                    df['Y_i'] = df['aktual'] + df['D_z'] / 2
                    df['YY_i'] = df['aktual'] - df['D_z'] / 2
            # Menampilkan hasil x_i, xx_i, Y_i, YY_i
                    selected_columns = ['aktual', 'fuzzy', 'X_i', 'XX_i', 'Y_i', 'YY_i']
                    print(df[selected_columns])
            #menampilkan panjang interval serta nilai tengah
                    print('panjang interval :',l)
                    print("NILAI TENGAH")
                    for i, midpoint in enumerate(midpoints):
                      print(f"u_{i+1}: {midpoint}")
# Menghitung Prediksi S. R. Singh
            def predict_sr_singh(row, intervals, l):
              for i, interval in enumerate(intervals):
                if (interval[0] <= row['Y_i'] < interval[1]) or (interval[0] <= row['YY_i']
            < interval[1]):  # Use parentheses for clarity
                    midpoint = (interval[0] + interval[1]) / 2
                    return midpoint - (1/4 * l)
              for i, interval in enumerate(intervals):
                if (interval[0] <= row['X_i'] < interval[1]) or (interval[0] <= row['XX_i']
            < interval[1]):  # Use parentheses for clarity
```

Lampiran 25 (Lanjutan)

```
midpoint = (interval[0] + interval[1]) / 2
            return midpoint + (1/4 * l)
           if (interval[0] > row['X_i'] > interval[1]) or (interval[0] > row['XX_i'] >
           interval[1]):  # Use parentheses for clarity
            midpoint = (interval[0] + interval[1]) / 2
            return midpoint
# Terapkan fungsi ke setiap baris untuk menghasilkan prediksi
        df['prediksi_singh'] = df.apply(predict_sr_singh, axis=1, args=(intervals, l))
        # Apply the function to create the predictions
# Menampilkan hasil prediksi singh
        selected_columns = ['aktual', 'fuzzy', 'prediksi_singh']
        print(df[selected_columns])
```

Lampiran 6 *Coding* Uji Akurasi pada *Fuzzy Time Series Markov Chain* serta *S. R. Singh*

```
#Markov Chain
        # menangani nilai NaN sebelum melakukan perhitungan
                df_clean = df.dropna(subset=['aktual', 'prediksi_markov'])  # Hapus
                baris dengan NaN di kolom yang relevan
                # Menghitung Mean Absolute Error (MAE)
                n = len(df)
                mae_markov = (1/n) * np.sum(np.abs(df['aktual'] -
                df['prediksi_markov']))
                print("MAE:", mae_markov)
                # Menghitung MAPE
                n = len(df)
                mape_markov = (1/n) * np.sum(np.abs(df['aktual'] -
                df['prediksi_markov']) / df['aktual']) * 100
                print("MAPE:", mape_markov, "%")
        import numpy as np
        from sklearn.metrics import mean_squared_error
                # Menghitung RMSE
                rmse_markov = np.sqrt((1/n) * np.sum((df['aktual'] -
                df['prediksi_markov'])**2))
                print("RMSE:", rmse_markov)
#S.R.Singh
        # Menghitung Mean Absolute Error (MAE)
                n = len(df)
                mae_singh = (1/n) * np.sum(np.abs(df['aktual'] -
                df['prediksi_singh']))
                print("MAE:", mae_singh)
        # Menghitung MAPE
n = len(df)
```

Lampiran 26 (Lanjutan)

```
        mape_singh = (1/n) * np.sum(np.abs(df['aktual'] -
        df['prediksi_singh']) / df['aktual']) * 100
        print("MAPE:", mape_singh, "%")
# Menghitung RMSE
        rmse_singh = np.sqrt((1/n) * np.sum((df['aktual'] -
        df['prediksi_singh'])**2))
        print("RMSE:", rmse_singh)
# Membuat DataFrame untuk membandingkan akurasi
comparison_df = pd.DataFrame({
   'Metode': ['FTS Markov Chain', 'FTS S. R. Singh'],
   'MAE': [mae_markov, mae_singh],  # gunakan hasil perhitungan
mae_markov
   'MAPE': [mape_markov, mape_singh], # gunakan hasil perhitungan
mape_markov
   'RMSE': [rmse_markov, rmse_singh] # gunakan hasil perhitungan
rmse_markov
})
# Menampilkan DataFrame perbandingan
print(comparison_df)
```

Lampiran 7 Coding Perhitungan Prediksi Bulan Selanjutnya dengan *S. R. Singh*

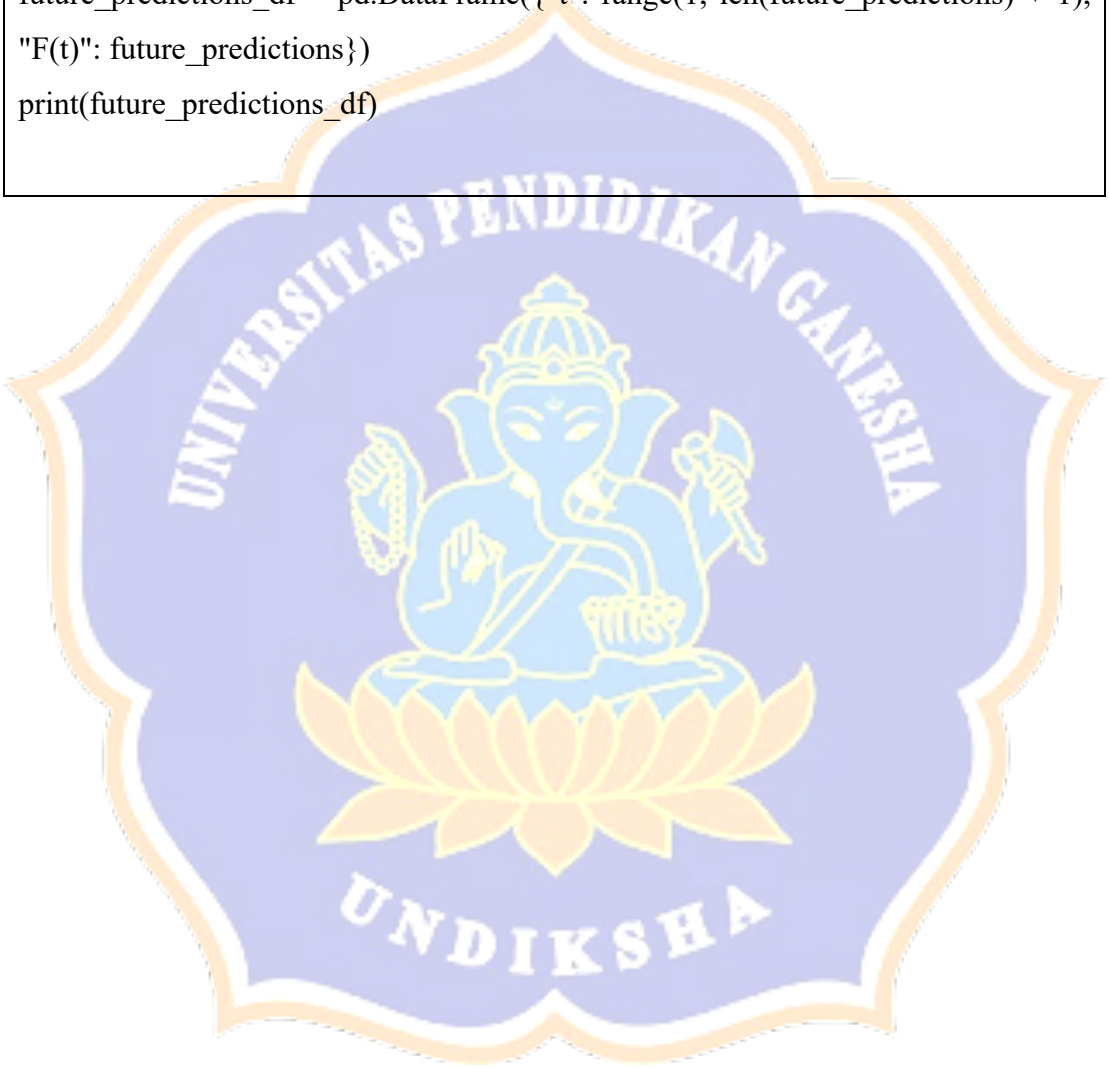
```
# ... (Kode sebelumnya tetap sama)
# Fungsi untuk memprediksi nilai inflasi bulan selanjutnya menggunakan metode yang paling akurat (dallam kasus ini, S. R. Singh)
def predict_future_sr_singh(df, intervals, l, months_ahead):
  """
        Memprediksi nilai masa depan menggunakan metode S. R. Singh.
        Args:
        df: DataFrame yang berisi data historis.
        intervals: Daftar interval
       I: Panjang setiap interval.
        months_ahead: Jumlah bulan yang akan diprediksi.
        Returns:
        list: Daftar prediksi untuk bulan mendatang.
  """
  future_predictions = []
  for i in range(bulan):
    last_row = df.iloc[-1]
    # Hitung D_z untuk baris terakhir
        D_z = np.abs(np.abs(last_row['aktual'] - df['aktual'].iloc[-2]) - np.abs(df['aktual'].iloc[-2] - df['aktual'].iloc[-3]))
    # Hitung X_i, XX_i, Y_i, YY_i untuk baris terakhir
    X_i = last_row['aktual'] + D_z
    XX_i = last_row['aktual'] - D_z
    Y_i = last_row['aktual'] + D_z / 2
    YY_i = last_row['aktual'] - D_z / 2
```

Lampiran 27 (Lanjutan)

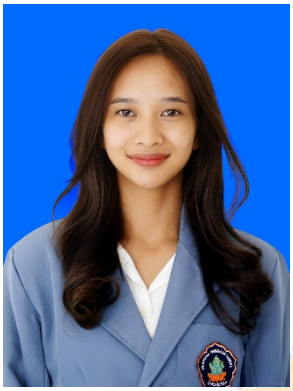
```
  # tampilkan hasil
    print(f"D_z: {D_z}")
    print(f"X_i: {X_i}")
    print(f"XX_i: {XX_i}")
    print(f"Y_i: {Y_i}")
    print(f"YY_i:{YY_i}")
    # Memprediksi nilai berikutnya
    for j, interval in enumerate(intervals):
      if (interval[0] <= Y_i < interval[1]) or (interval[0] <= YY_i < interval[1]):
        midpoint = (interval[0] + interval[1]) / 2
        prediction = midpoint - (1/4 * l)
        break
    else:
      for j, interval in enumerate(intervals):
        if (interval[0] <= X_i < interval[1]) or (interval[0] <= XX_i < interval[1]):
          midpoint = (interval[0] + interval[1]) / 2
          prediction = midpoint + (1/4 * l)
          break
        if (interval[0] > X_i > interval[1]) or (interval[0] > XX_i > interval[1]):
          midpoint = (interval[0] + interval[1]) / 2
          prediction = midpoint
    future_predictions.append(prediction)
    # Perbarui DataFrame untuk prediksi berikutnya
    new_row = pd.DataFrame({'aktual': [prediction]}, index=[len(df)])
    df = pd.concat([df, new_row])
  return future_predictions
```

Lampiran 27 (Lanjutan)

```
# Prediksi 1 bulan ke depan
bulan = 1
future_predictions = predict_future_sr_singh(df, intervals, l, bulan)
# tampilkan hasil prediksinya
future_predictions_df = pd.DataFrame({"t": range(1, len(future_predictions) + 1),
"F(t)": future_predictions})
print(future_predictions_df)
```



RIWAYAT HIDUP



Ni Wayan Putri Surya Deanik lahir di Sebatu pada tanggal 26 Juni 2001. Penulis merupakan anak pertama dari tiga bersaudara, yang lahir dari pasangan suami istri I Made Pantri dan Ni Wayan Purni. Penulis berkebangsaan Indonesia dan beragama Hindu. Kini penulis beralamat di Jalan Raya Bilukan, Sebatu, Kabupaten Gianyar, Bali. Penulis menyelesaikan Pendidikan dasar di SD No. 4 Sebatu pada tahun 2014. Kemudian melanjutkan Pendidikannya di SMP Negeri 1 Tegallalang dan lulus pada tahun 2017. Penulis lalu melanjutkan Pendidikannya di SMA Negeri 1 Sukawati dan lulus pada tahun 2020. Selanjutnya, penulis melanjutkan jenjang perguruan tinggi ke Program Studi S1 Matematika di Universitas Pendidikan Ganesha. Adapun Riwayat organisasi penulis selama menempuh Pendidikan di Universitas Pendidikan Ganesha, yakni sebagai Bendahara II pada kepengurusan HMJ Matematika masa bakti 2021/2022, dan Pengurus HMJ Matematika Masa bakti 2022/2023 sebagai Bendahara I. Penulis juga bergabung sebagai relawan pengajar di Taman Cerdas Ganesha. Pada semester akhir di tahun 2024, penulis menyelesaikan skripsi yang berjudul **"Analisis Perbandingan Akurasi Metode *Fuzzy Time Series Markov Chain* dan *S. R. Singh* dalam Memprediksi Laju Inflasi Kabupaten Buleleng"**.