BAB IV

HASIL DAN PEMBAHASAN

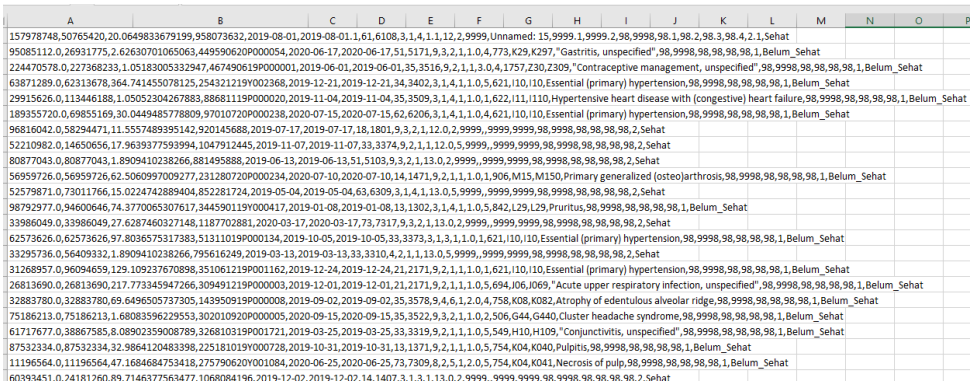
4.1 Hasil

Berdasarkan metodologi yang telah dirancang pada kasus prediksi kepulangan peserta BPJS menggunakan metode *decision tree* yang terdiri dari beberapa proses antara nya adalah sebagai berikut.

4.1.1 Data Collection

1. Menyiapkan dataset

Data yang akan diolah merupakan data dalam bentuk file CSV yang diperoleh dari Badan Penyelenggara Jaminan Kesehatan (BPJS). Data yang diolah dalam penelitian ini terdiri dari atas 10000 baris dan 26 *column*, Adapun dataset tersebut sebagai berikut.



A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
157978748,50765420,20.0649833679199,958073632,2019-08-01,2019-08-01.1,61,6108,3,1,4,1.1,12,2,9999,Unnamed: 15,9999.1,9999.2,98,9998,98.1,98.2,98.3,98.4,2.1,Sehat														
95085112.0,26931775,2.62630701065063,449590620P000054,2020-06-17,2020-06-17,51,5171,9,3,2,1,1.0,4,773,K29,K297,"Gastritis, unspecified",98,9998,98,98,98,98,1,Belum_Sehat														
224470578.0,227368233,1.05183005332947,467490619P000001,2019-06-01,2019-06-01,35,3516,9,2,1,1,3.0,4,1757,Z30,Z309,"Contraceptive management, unspecified",98,9998,98,98,98,98,1,Belum_Sehat														
63871289.0,62313678,364.741455078125,254321219Y002368,2019-12-21,2019-12-21,34,3402,3,1,4,1,1.0,5,621,I10,I10,Essential (primary) hypertension,98,9998,98,98,98,98,1,Belum_Sehat														
29915626.0,113446188,1.05052304267883,88681119P000020,2019-11-04,2019-11-04,35,3509,3,1,4,1,1.0,1,622,I11,I110,Hypertensive heart disease with (congestive) heart failure,98,9998,98,98,98,98,1,Belum_Sehat														
189355720.0,69855169,30.0449485778809,97010720P000238,2020-07-15,2020-07-15,62,6206,3,1,4,1,1.0,4,621,I10,I10,Essential (primary) hypertension,98,9998,98,98,98,98,1,Belum_Sehat														
96816042.0,58294471,11.5557489395142,920145688,2019-07-17,2019-07-17,18,1801,9,3,2,1,12.0,2,9999,,9999,9999,98,9998,98,98,98,98,2,Sehat														
52210982.0,14650656,17.9639377593994,1047912445,2019-11-07,2019-11-07,33,3374,9,2,1,1,12.0,5,9999,,9999,9999,98,9998,98,98,98,98,2,Sehat														
80877043.0,80877043,1.89094102382666,881495888,2019-06-13,2019-06-13,51,5103,9,3,2,1,13.0,2,9999,,9999,9999,98,9998,98,98,98,98,2,Sehat														
56959726.0,56959726,62.5060997009277,231280720P000234,2020-07-10,2020-07-10,14,1471,9,2,1,1,1.0,1,906,M15,M150,Primary generalized (osteo)arthrosis,98,9998,98,98,98,98,1,Belum_Sehat														
52579871.0,73011766,15.0224742889404,852281724,2019-05-04,2019-05-04,63,6309,3,1,4,1,13.0,5,9999,,9999,9999,98,9998,98,98,98,98,2,Sehat														
98792977.0,94600646,74.3770065307617,344590119Y000417,2019-01-08,2019-01-08,13,1302,3,1,4,1,1.0,5,842,L29,L29,Pruritus,98,9998,98,98,98,98,1,Belum_Sehat														
33986049.0,33986049,27.6287460327148,1187702881,2020-03-17,2020-03-17,73,7317,9,3,2,1,13.0,2,9999,,9999,9999,98,9998,98,98,98,98,2,Sehat														
62573626.0,62573626,97.8036575317383,51311019P000134,2019-10-05,2019-10-05,33,3373,3,1,3,1,1.0,1,621,I10,I10,Essential (primary) hypertension,98,9998,98,98,98,98,1,Belum_Sehat														
33295736.0,56409332,1.8909410238266,795616249,2019-03-13,2019-03-13,33,3310,4,2,1,1,13.0,5,9999,,9999,9999,98,9998,98,98,98,98,2,Sehat														
31268957.0,96094659,129.109237670898,351061219P001162,2019-12-24,2019-12-24,21,2171,9,2,1,1,1.0,1,621,I10,I10,Essential (primary) hypertension,98,9998,98,98,98,98,1,Belum_Sehat														
26813690.0,26813690,217.773345947266,309491219P000003,2019-12-01,2019-12-01,21,2171,9,2,1,1,1.0,5,694,J06,J069,"Acute upper respiratory infection, unspecified",98,9998,98,98,98,98,1,Belum_Sehat														
32883780.0,32883780,69.6496505737305,143950919P000008,2019-09-02,2019-09-02,35,3578,9,4,6,1,2.0,4,758,K08,K082,Atrophy of edentulous alveolar ridge,98,9998,98,98,98,98,1,Belum_Sehat														
75186213.0,75186213,1.68083596229553,302010920P000005,2020-09-15,2020-09-15,35,3522,9,3,2,1,1.0,2,506,G44,G440,Cluster headache syndrome,98,9998,98,98,98,98,1,Belum_Sehat														
61717677.0,38867585,8.089023590087890,326810319P001721,2019-03-25,2019-03-25,33,3319,9,2,1,1,1.0,5,549,H10,H109,"Conjunctivitis, unspecified",98,9998,98,98,98,98,1,Belum_Sehat														
87532334.0,87532334,32.9864120483398,225181019Y000728,2019-10-31,2019-10-31,13,1371,9,2,1,1,1.0,5,754,K04,K040,Pulpitis,98,9998,98,98,98,98,1,Belum_Sehat														
11196564.0,11196564,47.1684684753418,275790620Y001084,2020-06-25,2020-06-25,73,7309,8,2,5,1,2.0,5,754,K04,K041,Necrosis of pulp,98,9998,98,98,98,98,1,Belum_Sehat														
60393451.0,24181260,89.7146377563477,1068084196,2019-12-02,2019-12-02,14,1407,3,1,3,1,13.0,2,9999,,9999,9999,98,9998,98,98,98,98,2,Sehat														

Gambar 4.1. 1 Dataset

Data yang digunakan disediakan oleh BPJS Kesehatan yang berjumlah 4,056,898 dengan 26 column termasuk column target atau label yaitu kelas status kepulangan peserta BPJS dengan keterangan belum sehat dan sehat. [17] Adapun data sampel yang penulis gunakan sebanyak

10.000 data set, diharapkan dataset akan lebih maksimal dalam proses prediksi status kepulangan peserta BPJS. Data yang disediakan oleh BPJS Kesehatan diharapkan dapat menentukan pola status kepulangan peserta BPJS kesehatan yang sudah dinyatakan sehat atau belum sehat, sehingga hal ini dapat mengurangi bahkan mengatasi defisit anggaran BPJS kesehatan.

2. Menyiapkan Data dan Library

Bila dataset yang dibutuhkan sudah tersedia, tahap selanjutnya adalah menyiapkan library pada google colab untuk memudahkan dalam pengolahan data.

```
import pandas as pd
import numpy as np
import seaborn as sns
import statistics
from sklearn.preprocessing import OneHotEncoder
from sklearn.preprocessing import LabelEncoder
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix, accuracy_score, recall_score, precision_score
from sklearn.model_selection import train_test_split
from urllib.request import urlopen
import matplotlib.pyplot as plt
```

Gambar 4.1. 2 Import library yang digunakan

3. Mengimpor data ke google colab

Pada tahapan ini penulis akan mengimpor dataset yang telah didapat ke google colab untuk mempermudah pengolahan data.

```
'''
Kode ini akan mendowload file berukuran 600+ MB dari url, pastikan tidak menjalankannya berulangkali.
'''
url="https://dtc.darmajaya.ac.id/wp
content/uploads/train_fktp_converted.csv"
# # Download from URL
with urlopen(url) as file:
   content = file.read().decode()
save_as = "train_fktp.csv"
# Save to file
with open(save_as, 'w') as download:
   download.write(content)
```

Gambar 4.1. 3 Impor dataset

4. Mengetahui informasi dataset

Sebelum melakukan analisis data tahapan selanjutnya menampilkan informasi terkait jumlah kolom, tipe data dan jumlah seluruh data pada dataset.

```
df = pd.read_csv("train_fktp.csv")[:10000]
df.info( )
```

Output:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 26 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   157978748         10000 non-null  float64
 1   50765420          10000 non-null  int64
 2   20.0649833679199  10000 non-null  float64
 3   958073632         10000 non-null  object
 4   2019-08-01        10000 non-null  object
 5   2019-08-01.1      10000 non-null  object
 6   61                10000 non-null  int64
 7   6108              10000 non-null  int64
 8   3                 10000 non-null  int64
 9   1                 10000 non-null  int64
 10  4                 10000 non-null  int64
 11  1.1               10000 non-null  int64
 12  12                10000 non-null  float64
 13  2                 10000 non-null  int64
 14  9999              10000 non-null  int64
 15  Unnamed: 15       6560 non-null   object
 16  9999.1            10000 non-null  object
 17  9999.2            10000 non-null  object
 18  98                10000 non-null  int64
 19  9998              10000 non-null  int64
 20  98.1              10000 non-null  int64
 21  98.2              10000 non-null  int64
 22  98.3              10000 non-null  int64
 23  98.4              10000 non-null  int64
 24  2.1               10000 non-null  int64
 25  Sehat             10000 non-null  object
dtypes: float64(3), int64(16), object(7)
memory usage: 2.0+ MB
```

Gambar 4.1. 4 Informasi dataset

4.1.2 Exploratory Data Analysis

Pada tahapan EDA dilakukan analisis pada dataset untuk menampilkan data yang memiliki nilai null atau data kosong sehingga dapat dilakukan cleaning pada tahap *preprocessing.*

```
pd.isnull(df).sum()
```

Output:

```
nomor_peserta                                    0
nomor_keluarga                                   0
bobot                                            0
id_kunjungan_FKTP                                0
tanggal_datang_kunjungan_FKTP                    0
tanggal_pulang_kunjungan_FKTP                    0
provinsi_FKTP                                    0
kabupaten_kota_FKTP                              0
kepemilikan_FKTP                                 0
jenis_FKTP                                       0
tipe_FKTP                                        0
tingkat_pelayanan_FKTP                           0
jenis_poli_FKTP                                  0
segmen_peserta_saat_akses_layanan_FKTP           0
kode_dan_nama_diagnosis_ICD_10_3_digit           0
kode_diagnosis_ICD_10_3_digit                 3440
kode_diagnosis_beragam_3_5_digit                 0
nama_diagnosis_berasal_dari_kode_diagnosis_FKP15 0
provinsi_faskes_tujuan_rujukan                   0
kabupaten_kota_faskes_tujuan_rujukan             0
kepemilikan_faskes_tujuan_rujukan                0
jenis_faskes_tujuan_rujukan                      0
tipe_faskes_tujuan_rujukan                       0
poli_faskes_tujuan_rujukan                       0
jenis_kunjungan_FKTP                             0
kelas_status_pulang_peserta                      0
dtype: int64
```

Gambar 4.1. 5 Tahap EDA

4.1.3 Preprocessing

Berdasarkan Hasil Observasi Exploratory Data Analysis, diperlukannya beberapa preprocessing terlebih dahulu sebelum data dimasukan ke model untuk memprediksi kelas status pulang peserta. Adapun tahapan preprocessing yang dibutuhkan sebagai berikut.

a. Mengisi nilai null pada kolom kode_diagnosis_ICD_10_3_Digit

```
1 df['kode_diagnosis_ICD_10_3_digit'].fillna(df['kode_diagnosis_ICD_10_3_digit'].mode()[0], inplace = True)
```

mengecek nilai null pada kolom jenis_poli & kode_diagnosis_ICD_10_3_digit

```
[ ]  1 pd.isnull(df).sum()
```

```
nomor_peserta                                  0
nomor_keluarga                                 0
bobot                                          0
id_kunjungan_FKTP                              0
tanggal_datang_kunjungan_FKTP                  0
tanggal_pulang_kunjungan_FKTP                  0
provinsi_FKTP                                  0
kabupaten_kota_FKTP                            0
kepemilikan_FKTP                               0
jenis_FKTP                                     0
tipe_FKTP                                      0
tingkat_pelayanan_FKTP                         0
jenis_poli_FKTP                                0
segmen_peserta_saat_akses_layanan_FKTP         0
kode_dan_nama_diagnosis_ICD_10_3_digit         0
kode_diagnosis_ICD_10_3_digit                  0
kode_diagnosis_beragam_3_5_digit               0
nama_diagnosis_berasal_dari_kode_diagnosis_FKP15    0
provinsi_faskes_tujuan_rujukan                 0
kabupaten_kota_faskes_tujuan_rujukan           0
kepemilikan_faskes_tujuan_rujukan              0
jenis_faskes_tujuan_rujukan                    0
tipe_faskes_tujuan_rujukan                     0
poli_faskes_tujuan_rujukan                     0
jenis_kunjungan_FKTP                           0
kelas_status_pulang_peserta                    0
dtype: int64
```

Gambar 4.1. 6 Cleaning data

b. Pembuatan Kolom baru bernama lama_kunjungan

```
[ ]  1 bpjs_data_preprocess['tanggal_datang_kunjungan_FKTP'] = pd.to_datetime(bpjs_data_preprocess['tanggal_datang_kunjungan_FKTP'])
     2 bpjs_data_preprocess['tanggal_pulang_kunjungan_FKTP'] = pd.to_datetime(bpjs_data_preprocess['tanggal_pulang_kunjungan_FKTP'])
     3
     4 # Insert column 'lama_kunjungan' on index 1
     5 bpjs_data_preprocess.insert(1, "lama_kunjungan", bpjs_data_preprocess['tanggal_pulang_kunjungan_FKTP'] - bpjs_data_preprocess['tanggal_datang_kunjungan_FKTP'], True)
```

Mengecek kolom lama_kunjungan

```
[ ]  1 bpjs_data_preprocess
```

	nomor_peserta	lama_kunjungan	nomor_keluarga	bobot	id_kunjungan_FKTP	tanggal_datang_kunjungan_FKTP	tanggal_pulang_kunjungan_FKTP	provinsi_FKTP	kabupaten_kota_FKTP
0	95085112.0	0 days	26931775	2.626307	449590620P000054	2020-06-17	2020-06-17	51	5171
1	224470578.0	0 days	227368233	1.051830	467490619P000001	2019-06-01	2019-06-01	35	3516
2	63871289.0	0 days	62313678	364.741455	254321219Y002368	2019-12-21	2019-12-21	34	3402
3	29915626.0	0 days	113446188	1.050523	88681119P000020	2019-11-04	2019-11-04	35	3509
4	189355720.0	0 days	69855169	30.044949	97010720P000238	2020-07-15	2020-07-15	62	6206
...	...	...	...	...	...	...	...	...	...
9995	33425625.0	0 days	64990213	189.829437	400020819P000868	2019-08-13	2019-08-13	32	3201
9996	69400709.0	0 days	69400709	40.445122	49401119P000262	2019-11-12	2019-11-12	33	3305
9997	2157662.0	0 days	2157662	180.374741	317170319Y001247	2019-03-20	2019-03-20	35	3516
9998	3808844.0	0 days	3808844	274.501556	404660719Y000267	2019-07-06	2019-07-06	33	3310
9999	58711861.0	0 days	17738706	3.571777	231630920P000060	2020-09-14	2020-09-14	14	1471

10000 rows × 27 columns

Gambar 4.1. 7 Column lama_kunjungan

c. Penghapusan kolom yang tidak berpengaruh pada kelas status kepulangan peserta BPJS

```
bpjs_data_preprocess = df.drop(['nomor_peserta',  'nomor_keluarga',
                                'id_kunjungan_FKTP',
                                'tanggal_datang_kunjungan_FKTP',
                                'tanggal_pulang_kunjungan_FKTP'
                            ], axis = 1)
```

Gambar 4.1. 8 Kode drop column

d. Mengubah nilai pada kolom kelas_status_pulang_peserta yaitu nilai 0: belum sehat dan 1: sehat

```
bpjs_data_preprocess["kelas_status_pulang_peserta"] = np.where (bpjs
_data_preprocess["kelas_status_pulang_peserta"]== 'Belum_Sehat', 0, 1)
```

Gambar 4.1. 9 Kode ubah nilai column label

e. Mengubah tipe data kolom bobot dan jenis_poli_FKTP menjadi integer

```
#mengubah type data
bpjs_data_preprocess['jenis_poli_FKTP'] = bpjs_ data_preprocess
['jenis_poli_FKTP'] . astype(int)
bpjs_data_preprocess['bobot'] = bpjs_data_preprocess ['bobot'] .astype (int)
```

Gambar 4.1. 10 Kode ubah tipe data

4.1.3.1 Hasil Preprocessing

Hasil preprocessing berupa data yang telah dihilangkan atribut – atribut yang tidak diperlukan dan hanya menyertakan atribut yang diperlukan dalam proses data mining sehingga proses prediksi akan lebih efisien terhadap data yang lebih bersih.

Tabel 4.1.3. 1 Data set hasil preprocessing

lama_kunjungan	provinsi_FKTP	kabupaten_kota_FKTP	kepemilikan_FKTP
0	51	5171	9
0	35	3516	9
0	34	3402	3
0	35	3509	3
0	62	6206	3
0	18	1801	9
0	33	3374	9
0	51	5103	9
0	14	1471	9
0	63	6309	3
0	13	1302	3
0	73	7317	9
0	33	3373	3
0	33	3310	4
0	21	2171	9

poli_faskes_tujuan_rujukan	jenis_kunjungan_FKTP	kelas_status_pulang_peserta
98	1	belum sehat
98	1	belum sehat
98	1	belum sehat
98	1	belum sehat
98	1	belum sehat
98	2	sehat
98	2	sehat
98	2	sehat
98	1	belum sehat
98	2	sehat
98	1	belum sehat
98	2	sehat
98	1	belum sehat

98 2 sehat

4.2.3.1 Proses Algoritma C4.5

Langkah-langkah dalam pembentukan pohon Keputusan pada prediksi status kepulangan peserta BPJS menggunakan metode decision tree dilakukan sesuai pada data tabel 4.1.3.1, berikut perhitungan algoritma C4.5 :

NILAI ENTROPY DAN GAIN UNTUK MENENTUKAN SIMPUL AKAR

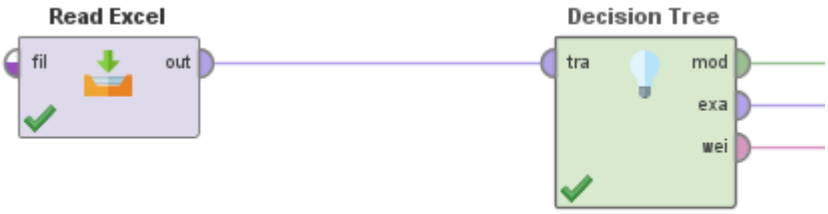
Atribut	Nilai	Jumlah Kasus	Belum_sehat	Sehat	Entrophy	Gain
Total		9	6	3	0,975404595	
Provinsi_FKTP						0,325134865
	Bali	3	3	0	0	
	Jawa Timur	3	2	1	0,975404595	
	Riau	3	1	2	0,975404595	
Kepemilikan_FKTP						0,308737929
	Swasta	6	3	3	1	
	PemKab	3	3	0	0	
Jenis_FKTP						0,325134865
	Dokter Umum	3	1	2	0,975404595	
	Klinik Pratama	3	2	1	0,975404595	
	Puskesmas	3	3	0	0	
Tipe_FKTP						0,325134865
	Dokter Praktek Perorang	3	1	2	0,975404595	
	Klinik Non Rawat Inap	3	2	1	0,975404595	
	Rawat Inap	3	3	0	0	
Jenis_Poli FKTP						0,975404595
	Poli_Umum	5	5	0	0	
	Poli_KIA	1	1	0	0	
	Poli_Gigi&Mulut	3	0	3	0	
eserta_Saat_Akses_Layanan_FKTP						0,753182373
	PBPU	3	3	0	0	
	PPU	2	1	1	1	
	Bukan pekerja	2	2	0	0	
	PBI APBN	2	0	2	0	
pe_Faskes_Tujuan_Rujukan						0,14121861
	Khusus Jiwa	5	4	1	0,916292016	
	Khusus Ibu & anak	3	2	1	0,975404595	
	Khusus gigi dan mulut	1	0	1	0	
Jenis_Kunjungan_FKTP						0,975404595
	Kunjungan sakit	6	6	0	0	
	Kunjungan sehat	3	0	3	0	

Gambar 4.1. 11 Perhitungan Entropy

4.3.3.1 Menentukan Nilai Atribut

Pada tahapan ini penelitian akan dilakukan menggunakan *software rapid miner* dengan metode decision tree. Untuk tahapan berikut penulis menggunakan operator *read excel* yang berfungsi untuk

membaca file yang diolah, kemudian menghubungkan dengan operator *decision tree* untuk membuat data yang diolah menghasilkan pohon keputusan.



Gambar 4.1. 12 Proses Awal Rapid Miner

4.4.3.1 *Example Set*

Example set merupakan dataset yang telah terupload di aplikasi *Rapid Miner*. Berikut adalah dataset yang telah siap untuk diproses dengan aplikasi *Rapid Miner*.

Row No.	kelas_statu...	provinsi_FK...	kepemilikan...	jenis_FKTP	tipe_FKTP	jenis_poli_F...	segmen_pe...	tipe_faskes...	jenis_kunju...
1	Belum_sehat	Bali	Swasta	Dokter_umum	Dokter Prakte...	Poli_umum	PBPU	Khusus_Jiwa	Kunjungan_s...
2	Belum_sehat	Bali	Swasta	Klinik_pratama	Klinik Non Ra...	Poli_KIA	PBPU	Khusus_Jiwa	Kunjungan_s...
3	Belum_sehat	Bali	PemKab	Puskesmas	Rawat_inap	Poli_umum	PPU	Khusus_Ibu&...	Kunjungan_s...
4	Belum_sehat	Jawa Timur	PemKab	Puskesmas	Rawat_inap	Poli_umum	Bukan_pekerja	Khusus_Ibu&...	Kunjungan_s...
5	Belum_sehat	Jawa Timur	PemKab	Puskesmas	Rawat_inap	Poli_umum	PBPU	Khusus_Jiwa	Kunjungan_s...
6	Sehat	Jawa Timur	Swasta	Dokter_umum	Dokter Prakte...	Poli_Gigi & M...	PBI APBN	Khusus_Jiwa	Kunjungan_s...
7	Sehat	Riau	Swasta	Klinik_pratama	Klinik Non Ra...	Poli_Gigi & M...	PPU	Khusus_Ibu&...	Kunjungan_s...
8	Sehat	Riau	Swasta	Dokter_umum	Dokter Prakte...	Poli_Gigi & M...	PBI APBN	Khusus Gigi ...	Kunjungan_s...
9	Belum_sehat	Riau	Swasta	Klinik_pratama	Klinik Non Ra...	Poli_umum	Bukan_pekerja	Khusus_Jiwa	Kunjungan_s...

Gambar 4.1. 13 Example Set

4.5.3.1 Description

Description menjelaskan hasil dari klasifikasi data yang telah diolah pada *Rapid Miner*. Hasil dari description dapat dilihat pada gambar berikut.

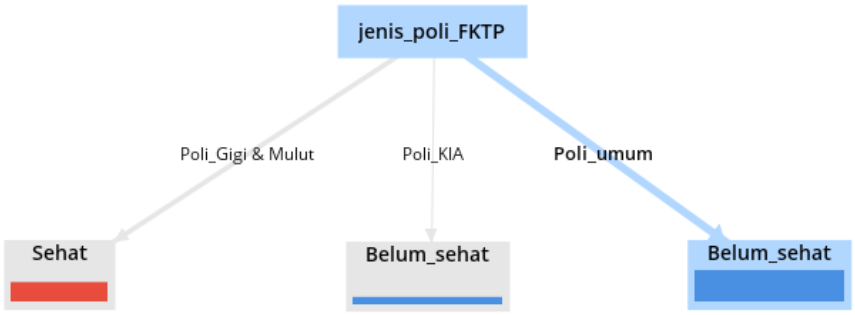
Tree

```
jenis_poli_FKTP = Poli_Gigi & Mulut: Sehat {Belum_sehat=0, Sehat=3}
jenis_poli_FKTP = Poli_KIA: Belum_sehat {Belum_sehat=1, Sehat=0}
jenis_poli_FKTP = Poli_umum: Belum_sehat {Belum_sehat=5, Sehat=0}
```

Gambar 4.1. 14 Description dataset

4.6.3.1 Pohon Keputusan

Pohon Keputusan atau *graph view* menunjukan hasil dari percabangan yang dapat dihasilkan kesimpulan. Berikut adalah hasil dari *decision tree.*



Gambar 4.1. 15 Hasil Decision Tree

4.1.4 Modelling

a. Menentukan variabel x dan y

Tahap selanjutnya sebelum memodelkan menggunakan decision tree, hal yang dilakukan adalah menentukan variabel x dan y.

```
data_train= data_model[['lama_kunjungan', 'bobot', 'provinsi_FKTP','kabupaten_kota_FKTP','kepemilikan_FKTP','jenis_FKTP','
data_label = data_model['kelas_status_pulang_peserta'].values
```

Gambar 4.1. 16 Menentukan variabel

b. Melakukan *split* data training dan testing

Tahap selanjutnya membagi data menjadi dua bagian yaitu data training dan data testing Dimana proporsi yang digunakan pada penelitian ini adalah 20:80. Rasio tersebut dimaksudkan untuk membagi data training sebesar 80% dan data testing sebesar 20%. Adapun *split* data training dan testing sebagai berikut.

```
# Membagi dataset menjadi data pelatihan dan data uji
```

```
X_train, X_test, y_train, y_test = train_test_split(data_train, data_label,
test_size=0.2, random_state=42)
```

```
from sklearn.preprocessing import StandardScaler
s_scaler = StandardScaler()
X_train = s_scaler.fit_transform(X_train.astype(int))
X_test = s_scaler.transform(X_test.astype(int))
```

Gambar 4.1. 17 Split data training testing

4.1.5 Evaluation

a. Klasifikasi menggunakan decision tree

Pada tahap evaluation dalam melakukan prediksi diperlukan klasifikasi data untuk menilai performa data, Adapun evaluasi menggunakan model algoritma C4.5 menggunakan kode berikut.

```
model = DecisionTreeClassifier()
model.fit (X_train, y_train)
# Membuat prediksi
y_pred_train = model.predict(X_train)
y_pred_test = model.predict(X_test)
```

Gambar 4.1. 18 Klasifikasi menggunakan decision tree

b. Menentukan akurasi

Setelah melakukan klasifikasi, tahap selanjutnya menentukan nilai akurasi algoritma C4.5 menggunakan kode berikut.

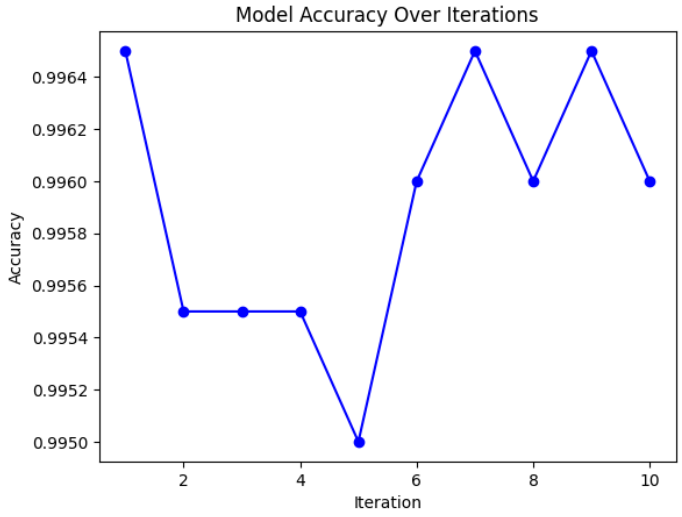
```
# Evaluasi performa model
accuracy_train = accuracy_score(y_train, y_pred_train)
accuracy_test = accuracy_score(y_test, y_pred_test)

print(f'Accuracy on training set: {accuracy_train}')
print(f'Accuracy on testing set: {accuracy_test}')
```

```
Accuracy on training set: 1.0
Accuracy on testing set: 0.996
```

Gambar 4.1. 19 Penentuan Akurasi

Dari model akurasi diperoleh sebesar 99%, sehingga dapat diartikan dari seluruh data yang diolah baik data training dan data testing, kemampuan algoritma C4.5 dalam melakukan prediksi adalah sebesar 99%. Nilai akurasi ini dianggap sudah sangat baik dalam melakukan prediksi dan dapat dijadikan acuan untuk dilakukan prediksi kepulangan peserta BPJS yang sudah layak sehat atau belum sehat.

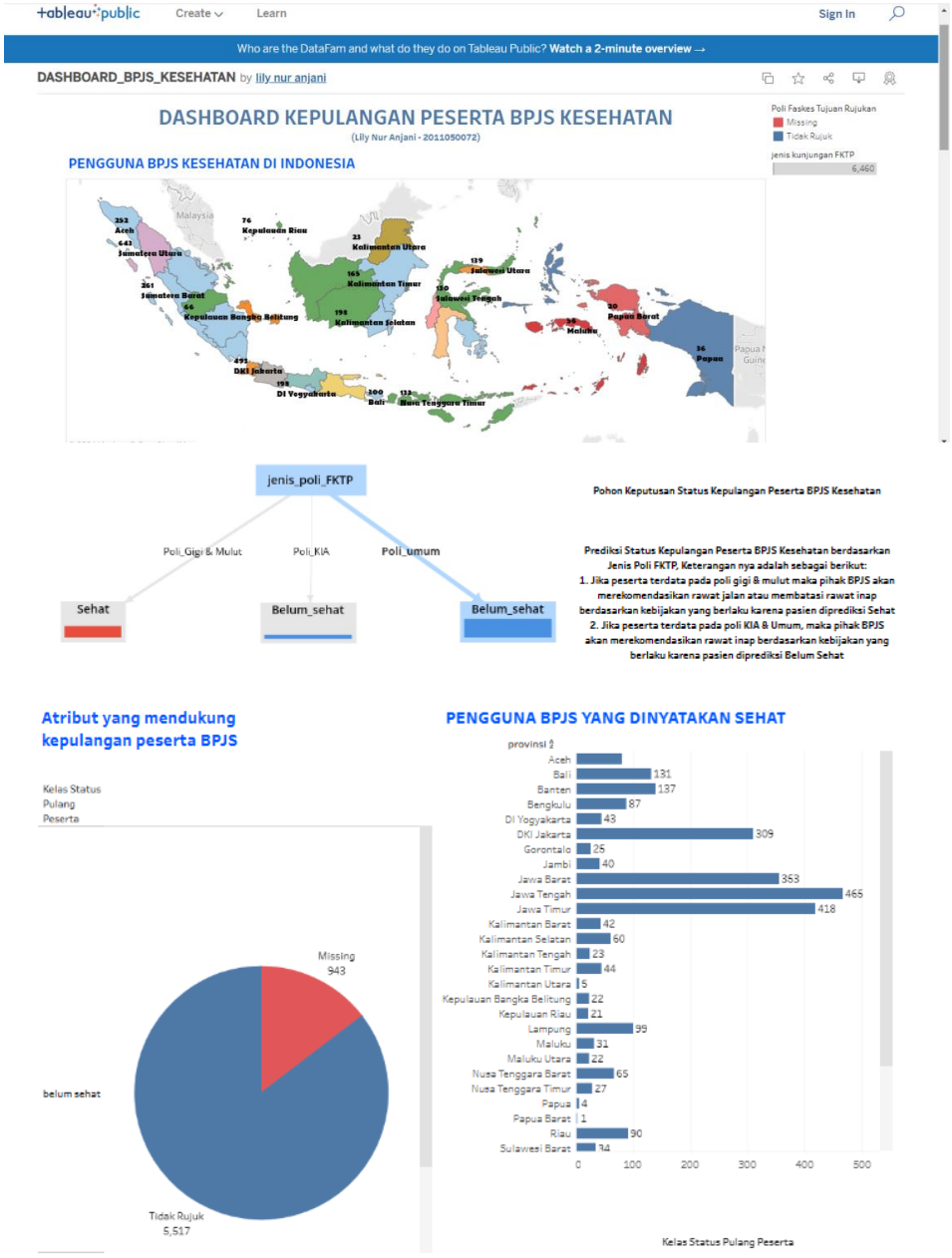


Gambar 4.1. 20 Performa akurasi

Berdasarkan visualisasi diatas maka dapat dilihat bahwa model dapat mengenali pola dataset dengan baik dan memiliki akurasi model mencapai nilai 0,996 atau 99%.

4.1.6 Visualisasi dashboard

Tahap visualiasi dashboard peneliti menggunakan dashboard tableu untuk mempermudah dalam visualisasi data.



Gambar 4.1. 21 Visualisasi Dashboard Pengguna BPJS

Berdasarkan visualisasi dashboard diatas, penulis menggunakan dashboard tableu untuk memvisualisasikan dataset BPJS kesehatan. Terdapat peta

provinsi yang menampilkan pengguna BPJS Kesehatan yang ada di Indonesia. Pada dashboard tableu penulis juga menampilkan atribut yang mendukung kepulangan peserta BPJS yaitu pada atribut jenis faskes tujuan rujukan , pada dataset ini terdapat 5.517 tidak rujuk dan 913 data missing serta dinyatakan belum sehat. Hal ini seharusnya dapat dianalisis dan dipantau lebih lanjut oleh pihak BPJS sehingga peserta yang dinyatakan tidak rujuk dan layak dinyatakan sehat dapat dipulangkan atau dilakukan rawat jalan. Sehingga anggaran BPJS dapat tersalurkan dengan tepat dan defisit anggaran dapat teratasi dengan baik.